UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Ingeniería Técnica en Informática de Gestión

Proyecto Fin de Carrera

Creación de un visor HTML5 para personas con discapacidad auditiva

Autor: D. Eduardo García Rodríguez

Director: Prof. Ángel García Crespo

Octubre, 2011

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

El Proyecto Fin de Carrera (PFC) de la titulación Ingeniería Técnica en Informática de Gestión denominado "Creación de un visor HTML5 para personas con discapacidad auditiva" consiste en la creación de un visor de vídeos HTML5 accesible para personas con discapacidad auditiva. Para ello, se utilizan subtítulos que se encuentran en un archivo XML siguiendo el estándar TT AF – DFXP.

Con la realización de este proyecto es posible adelantarse en el conocimiento del lenguaje HTML5 que de cada vez está siendo más utilizado y que va a ser en el futuro el lenguaje predominante, en detrimento del lenguaje Flash. Además, se tiene la motivación de desarrollar una aplicación accesible para personas con discapacidad auditiva, por lo que se les facilita la visualización de los vídeos.

1.1. DESCRIPCIÓN GENERAL DEL TRABAJO

El desarrollo del presente Proyecto Fin de Carrera se puede dividir en tres fases claramente diferenciadas:

1.1.1. PLANTEAMIENTO DEL PROBLEMA Y SOLUCIONES

Constituye la primera fase del proyecto. En ella se realiza un estudio acerca de la carencia del uso de subtítulos en formato TT AF – DFXP existente en los visores de vídeos en formato HTML5 y, por tanto, la necesidad de implementar los subtítulos en dicho formato para mejorar la accesibilidad de los visores de vídeos en HTML5. Además, se detallan las soluciones llevadas a cabo para solventar estos problemas encontrados.

1.1.2. ANÁLISIS Y DISEÑO

Durante esta fase del proyecto, se lleva a cabo el análisis de los requisitos que debe cumplir el software de la aplicación, tanto los requisitos funcionales como los no funcionales, siendo necesario que el conjunto de requisitos sea lo más completo, consistente y correcto posible. Una vez establecidos los requisitos, se pasa a la fase de diseño de la aplicación.

1.1.3. IMPLEMENTACIÓN Y PRUEBAS

Se trata de la última fase del proyecto. En ella, se implementa la aplicación, tomando como referencia el diseño del mismo realizado en la anterior fase del proyecto. Una vez implementado la aplicación, se realizan las correspondientes pruebas para verificar el correcto funcionamiento de la aplicación. Es necesario refinar la aplicación hasta conseguir cumplir la especificación de requisitos del mismo.

1.2. DESCRIPCIÓN DE LOS CAPÍTULOS DEL PFC

En el capítulo 2 se describe la finalidad del proyecto y, por tanto, los objetivos que se deben alcanzar a la hora de realizar el PFC.

En el capítulo 3 se describe el estado de la técnica donde se explica el estado en el que se encuentran en la actualidad tanto el lenguaje HTML5 como el estándar TT AF – DFXP.

En el capítulo 4 se describe la arquitectura del PFC donde se explican los pasos seguidos para la realización de la aplicación.

En el capítulo 5 se describe la planificación del PFC donde se detallan las tareas que componen el proyecto, el tiempo empleado en la realización de cada una de ellas, así como el presupuesto del proyecto.

En el capítulo 6 se describen las conclusiones obtenidas una vez terminado el PFC.

En el capítulo 7 se describen las futuras líneas/trabajos que se pueden realizar sobre el proyecto para mejorarle.

En el siguiente apartado se describen los anexos necesarios para ampliar la información y completar el presente documento.

Posteriormente, se describe el glosario de términos y acrónimos.

Finalmente, se describen las referencias tomadas para realizar la memoria.

2. OBJETIVOS

El cometido de este Proyecto Fin de Carrera es desarrollar un visor de vídeos en HTML5 accesible para personas con discapacidad auditiva, por tanto, se trata del objetivo principal.

Para alcanzar este objetivo, se han establecidos los siguientes subobjetivos:

1. Estudio de la situación actual del lenguaje HTML5 y del estándar TT AF – DFXP.

2. Diseño e implementación de un programa informático que permita el desarrollo de la aplicación deseada.

3. Evaluación de las capacidades de la aplicación y comprobación del cumplimiento de los objetivos planteados.

3. ESTADO DE LA TÉCNICA

En este capítulo se describe el estado en el que se encuentran en la actualidad tanto el lenguaje HTML5 como el estándar TT AF – DFXP. Este estudio ha sido realizado previamente al diseño de la aplicación.

En primer lugar, el estudio se centra en el lenguaje HTML5 y, posteriormente en el estándar TT AF – DFXP. En ambos casos, primero se estudian los antecedentes y, posteriormente, se estudian en el presente.

Mediante este estudio, se refuerzan los conceptos adquiridos previamente y se adquieren otros completamente desconocidos en un principio.

3.1. LENGUAJE HTML5

Esta nueva versión del lenguaje básico del Web proporciona mecanismos para simplificar el trabajo y facilitar la inclusión de elementos multimedia. El principal criterio de diseño de HTML5 ha sido el de resolver problemas prácticos, y con este objetivo adopta soluciones dirigidas a facilitar el trabajo en situaciones reales.

3.1.1. ANTECEDENTES

El lenguaje marcado de la World Wide Web ha sido siempre HTML. HTML fue diseñado principalmente como un lenguaje para describir semánticamente los documentos científicos, aunque su diseño general y las adaptaciones en los últimos años han permitido que sea utilizado para describir una serie de otros tipos de documentos.

El área principal que no ha sido tratado adecuadamente por HTML es un tema vago referido a las aplicaciones Web. Esta aplicación trata de corregir esta situación, mientras que al mismo tiempo la actualización de las especificaciones de HTML para hacer frente a las cuestiones planteadas en los últimos años.

Durante sus primeros cinco años (1990–1995), HTML pasó por una serie de revisiones y experimentó una serie de extensiones, todo organizado por primera vez en el CERN, y después en el IETF.

Con la creación de la W3C, el desarrollo de HTML ha cambiado el lugar de nuevo. Un primer intento fallido de extender HTML en 1995 conocido como HTML 3.0 dio paso a un enfoque más pragmático conocido como HTML 3.2, que fue completado en 1997. HTML4 siguió sus pasos, llegando a la finalización en 1998.

En este momento, los miembros del W3C decidieron dejar la evolución de HTML y, en cambio, comenzar a trabajar en un equivalente basado en XML, llamado XHTML. Este esfuerzo comenzó con una reformulación de HTML4 en XML, conocido como XHTML 1.0, que no añade nuevas características excepto la nueva serialización, y que fue completado en 2000. Después de XHTML 1.0, el enfoque de W3C se volvió a hacer más fácil para otros grupos de trabajo para extender XHTML, bajo la bandera de la Modularización de XHTML. Paralelamente a esto, el W3C también trabajó en un nuevo lenguaje que no era compatible con los anteriores lenguajes HTML y XHTML, llamándole XHTML 2.

En la época en que la evolución de HTML se detuvo en 1998, las partes de la API de HTML desarrolladas por los fabricantes de navegadores se especificaron y publicaron bajo el nombre de DOM Nivel 1 (en 1998) y DOM Nivel 2 Núcleo y DOM Nivel 2 HTML (empezando en 2000 y culminando en 2003). Estos esfuerzos entonces se agotaron, con algunas especificaciones DOM nivel 3 publicadas en 2004, pero el grupo de trabajo se cierra antes de que todos los proyectos de nivel 3 fuesen completados.

En 2003, la publicación de XForms, una tecnología que se posicionó como la siguiente generación de formularios Web, provocó un renovado interés en la evolución de HTML en sí mismo, en lugar de encontrar reemplazos para ello. Este interés nació de la constatación de que la implementación de XML como tecnología de la Web se limita a tecnologías totalmente nuevas (como RSS y más tarde Atom), en lugar de como un sustituto de las tecnologías existentes desplegadas (como HTML).

Una prueba de concepto para demostrar que era posible ampliar las formas HTML4 de ofrecer muchas de las características que introdujo XForms 1.0, sin requerir navegadores para implementar motores de representación que eran incompatibles con las páginas web HTML existentes, fue el primer resultado de este renovado interés. En esta temprana etapa, mientras que el proyecto ya estaba a disposición del público, y de entrada ya estaba siendo solicitado por todas las fuentes, la especificación estaba sólo bajo el derecho de autor de Opera Software.

La idea de que la evolución de HTML debe reabrirse fue probada en un taller del W3C en 2004, donde algunos de los principios que subyacen en el trabajo HTML5 (descrito más adelante), así como el mencionado primer borrador de la propuesta que cubre sólo las características de los formularios relacionados, fueron presentados a W3C conjuntamente por Mozilla y Opera. La propuesta fue rechazada sobre la base de que la propuesta estaba en conflicto con la dirección previamente elegida para la evolución de la Web, el personal del W3C y los miembros votaron a favor de continuar el desarrollo de sustitutos basados en XML en su lugar.

Poco después, Apple, Mozilla y Opera anunciaron conjuntamente su intención de seguir trabajando en el esfuerzo en el marco de un nuevo espacio llamado el WHATWG. Una lista de correo pública fue creada, y el proyecto fue trasladado al sitio WHATWG. Los derechos de autor fueron modificados posteriormente para ser propiedad conjunta de los tres proveedores, y que permitan la reutilización de la especificación.

El WHATWG se basó en varios principios fundamentales, en particular, que las tecnologías deben ser compatibles hacia atrás, que las especificaciones e implementaciones necesitan adaptarse incluso si esto significa cambiar la especificación en lugar de las implementaciones, y que las especificaciones deben ser lo suficientemente detalladas que las implementaciones se puede lograr interoperabilidad completa sin la ingeniería inversa entre sí.

Este último requisito, en particular, requiere que el alcance de la especificación HTML5 incluya lo que se había especificado anteriormente en tres documentos separados: HTML4, XHTML1 y HTML DOM2. También significó que se incluyan detalles más significativamente de lo que previamente se había considerado la norma.

En 2006, el W3C manifestó su interés por participar en el desarrollo de HTML 5, después de todo, y en 2007 formó un grupo de trabajo constituida para trabajar con el WHATWG en el desarrollo de la especificación HTML5. Apple, Mozilla y Opera permitieron al W3C publicar la especificación bajo el derecho de autor del W3C, manteniendo una versión con la licencia menos restrictiva en el sitio WHATWG.
Desde entonces, ambos grupos han estado trabajando juntos.

La especificación HTML publicada por el WHATWG no es idéntica a esta especificación. Las diferencias principales son que la versión WHATWG incluye características no incluidas en esta versión del W3C: algunas características han sido omitidas, ya que se consideran parte de las revisiones futuras de HTML, no HTML5; y otras características son omitidas porque en el W3C son publicadas especificaciones separadas. También hay algunas pequeñas diferencias.

Un documento separado ha sido publicado por el grupo de trabajo HTML del W3C para documentar las diferencias entre esta especificación y el lenguaje descrito en la especificación HTML4.

3.1.2. SINTAXIS Y CORRECCIÓN DE ERRORES

XHTML se creó para sustituir la sintaxis en la que se basa HTML: el SGML, cuya finalidad principal era facilitar la creación manual de documentos. HTML era un lenguaje poco riguroso y, como consecuencia, los documentos resultaban un tanto caóticos y no siempre cumplían con la sintaxis. Esto provocó que los navegadores se hicieran más complejos para poder asumir ambigüedades y equivocaciones, es decir, para poder representar los documentos aunque tuvieran errores.

Como las máquinas tienen ciertas dificultades para leer y manipular contenido etiquetado con la sintaxis de SGML, se propuso una sintaxis nueva basada en XML, un lenguaje derivado de SGML pero más comprensible para las máquinas. Así, al aplicar a HTML la sintaxis de XML, los ordenadores son capaces de manipular XHTML con facilidad y precisión.

La especificación de HTML5 no se adscribe a una sintaxis o a la otra, sino que admite ambas serializaciones: HTML y XHTML. De esta manera, los creadores de contenido pueden escoger entre un enfoque práctico aunque poco riguroso (sintaxis HTML) y una visión académica y estricta (sintaxis XHTML). Con el tiempo, el W3C ha acabado aceptando que XHTML sea una recomendación paralela que puede coexistir con HTML.

HTML5 no sólo define cómo se deben analizar los documentos, sino también cómo se deben interpretar si no son válidos o si están mal formados. Actualmente, los navegadores corrigen los errores de sintaxis de distinta manera, del modo en que a los fabricantes les resulta más práctico. HTML5 trata de poner fin a esa necesidad de ingeniería inversa de los navegadores, que compiten por definir cómo se deben subsanar los errores (Andersson, 2007; Keith, 2010).

3.1.3. DOM

Una de las novedades principales de HTML5 es la inclusión del DOM como fundamento del lenguaje. El DOM describe la estructura de un documento de acuerdo con el paradigma de la orientación a objetos. En otras palabras, define el conjunto de entidades que están presentes en un documento HTML y las acciones que pueden realizarse sobre ellas.

Hasta ahora, el DOM siempre se había tratado de forma separada; cada navegador lo implementaba según la particular interpretación que su fabricante hacía de él. En cambio, en HTML5, el DOM forma parte del estándar. Así se garantiza que los navegadores interpretarán adecuadamente la sintaxis de HTML y que al mismo tiempo implementarán las funciones del DOM que la sustentan.

Si el DOM está vinculado al lenguaje HTML, se evita el tener que desarrollar versiones distintas de una misma página para varios programas. El DOM se incluye mediante la extensión de sus API, a las que añade funciones nuevas. Esta ampliación de las capacidades del DOM permite ejecutar funciones sofisticadas que hasta ahora requerían el desarrollo de programas y componentes adicionales (Álvarez, 2010). Por ejemplo, *Google* prevé sustituir con mecanismos propios de HTML su extensión *Gears*, desarrollada para sincronizar contenidos para ser usados sin conexión.

Para describir la interfaz de los elementos del DOM con un lenguaje neutro, los autores de HTML5 han escogido el IDL. Este lenguaje tiene una sintaxis similar a C++, que proporciona métodos para definir conceptos asociados a la programación orientada a objetos: atributos, métodos, constantes, herencias, etc. Pero IDL presenta una desventaja importante: carece de mecanismos para especificar la jerarquía que debe haber entre elementos y las restricciones (obligatoriedad y valores posibles) que cabe aplicar a los atributos.

No obstante, los autores de HTML5 han escogido este lenguaje, en lugar de DTD o *XML Schema*, porque puede definir elementos y comportamientos sin depender de una sintaxis concreta, una tarea imposible para DTD *y XML Schema* (Korostov y Paramzin, 2010).

3.1.4. PRESENTACIÓN Y RECOGIDA DE INFORMACIÓN

HTML5 incluye elementos nuevos destinados a enriquecer la presentación de documentos. Son ejemplos de ello los elementos semánticos *article*, *header*, *hgroup*, *nav*, *section*, *aside* y *footer*. Con ellos se pretende evitar que los autores abusen del elemento *div* para delimitar partes de un documento. Los blogs y los sitios de noticias han influido en gran medida en esta evolución (Schafer, 2010).

Por otra parte, existe un medio para que el usuario pueda hacer llegar datos de entrada a un servidor: los formularios, que recogen información que después remiten a aplicaciones que se ejecutan en el servidor (CGI, API, JSP, *servlets* u otras interfaces). HTML5 define más de una docena de nuevos controles (*email*, *range*, *date*, *time*, *placeholder*, *autofocus*, etc.) que actúan, por fin, sin necesidad de utilizar *JavaScript*, un lenguaje de programación que no todos los usuarios tienen habilitado (Pilgrim, 2010).

HTML5 propone, pues, que sean los navegadores, y no los creadores de contenido, quienes faciliten la entrada y la validación de datos que tienen un patrón regular o están sometidos a restricciones. Así, el dolor de cabeza que supone para los diseñadores de páginas web el verificar el formato de direcciones electrónicas, intervalos de valores, términos de búsqueda, colores, fechas y horas, entre otros tipos de datos, se descarga ahora sobre los navegadores.

3.1.5. FLASH Y CONTENIDO MULTIMEDIA

Para incrustar contenido multimedia, HTML ya contaba con el elemento *object*, pero la nueva versión del estándar hace una propuesta más semántica. En el ámbito del multimedia, HTML5 incorpora directivas nuevas que actúan como contenedores de vídeo, gráficos vectoriales y audio.

Se espera que con los elementos *video* y *audio* los navegadores tengan la capacidad de presentar el contenido de forma nativa, es decir, sin requerir componentes externos como *Flash*. Al fin y al cabo, una tarea sencilla como representar contenido multimedia no tiene por qué quedar en manos de un entorno cerrado y propietario.

Este avance ha suscitado el debate sobre si HTML5 acabaría con *Flash*. Es cierto que la propuesta de HTML5 puede sustituir funciones que hasta ahora sólo eran posibles con *Flash*. Para reproducir multimedia no hace falta ningún mecanismo complejo, con que es razonable pensar que HTML5 desplazará a *Flash* como contenedor de vídeo y audio, pero no hay argumentos sólidos para pensar que *Flash* vaya a desaparecer (Allaire, 2010).

Al fin y al cabo, *Flash* no sólo sirve para mostrar contenido multimedia, sino que también tiene capacidad para resolver muchas otras tareas. El objetivo de la propuesta de HTML5 es que la reproducción de vídeo y audio sea más eficiente, consuma menos recursos y se pueda gestionar con un código abierto y transparente, y sin necesidad de instalar componentes adicionales.

Todo apunta a que los navegadores incorporarán estas capacidades y que muchas páginas web se modificarán para aprovecharlas. Pero de momento, aunque incluidas en el estándar, esas funcionalidades se encuentran en la primera etapa de su adopción y no permiten controlar el contenido multimedia en la misma medida que *Flash*.

De hecho, el popular depósito de vídeos *YouTube* ha declarado recientemente que HTML5 no cubre todas sus necesidades (Harding, 2010). Varios son los problemas: HTML5 no cuenta todavía con un formato de vídeo estándar -se disputan ese puesto *H.264*, *Ogg Theora* y *WebM*-, no sustenta el visionado a pantalla completa ni permite escoger la calidad de reproducción. Por lo tanto, aunque las novedades de HTML5 son un avance hacia los estándares abiertos, es razonable pensar que *Flash* continuará desempeñando un papel vital en la distribución de contenido multimedia.

3.1.6. PRESENTE Y FUTURO

HTML5 se presenta como un marco estable para el desarrollo de páginas web. Ha sido definido para durar muchos años gracias a diversos mecanismos que permiten extender el lenguaje con facilidad. La filosofía de HTML5 aboga además por los estándares abiertos, que son fundamentales para impulsar la innovación e introducir en la sociedad los beneficios de las nuevas tecnologías. Bien sabemos que los estándares abiertos tienen la capacidad de mejorar nuestras vidas, y de ello son ejemplo las nuevas funciones multimedia de HTML5.

Debido al uso extendido de determinados complementos, como *Flash*, las nuevas funciones multimedia de HTML5 no pueden aspirar a sustituirlos por completo. Pero facilitan otros métodos para presentar contenido multimedia. La intención es no depender tanto de los complementos externos y propietarios.
Con todo, el proyecto de HTML5 ha comenzado a ir un poco a la deriva.

Algunos navegadores han comenzado a exhibir desarrollos propietarios (Powers, 2010), lo cual amenaza con socavar la filosofía fundamental. Además, la aparente escisión entre el W3C y el WHATWG sugiere que HTML5 podría andar fuera del camino deseado, o podría emitir un estándar diferente del que la industria necesita (Huggers, 2010).

El W3C y los fabricantes de navegadores representan el futuro del Web. HTML5 debe cumplir con su promesa de favorecer un navegador estándar, abierto y único. Ésta debe ser la preocupación principal, más allá del dilatado debate, quizá excesivo, en torno a la reproducción de vídeo. Aún hay mucho por hacer en HTML5 antes de poder integrarse en todos los ámbitos del Web. Hay tiempo suficiente para volver al buen camino.

3.2. ESTÁNDAR TT AF – DFXP

El TT AF es un tipo de contenido que representa el texto sincronizado de los medios de comunicación con el propósito de intercambio entre sistemas de edición. Texto sincronizado es la información textual que es intrínsecamente o extrínsecamente asociada con la información de la sincronización.

El DFXP está destinado para ser utilizado con el propósito de la transcodificación o el intercambio de información de texto sincronizado entre los formatos contenidos en la distribución del patrimonio actualmente en uso para el subtitulado y las funciones de subtitulado.

Además de ser utilizado para el intercambio entre los formatos contenidos en la distribución del patrimonio, el contenido DFXP puede ser utilizado directamente como un formato de distribución.

3.2.1. ANTECEDENTES

El 16 de noviembre de 2006, el W3C publica una Recomendación Candidata para indicar que el documento se cree que es estable y fomentar la aplicación por parte de la comunidad de desarrolladores. El TT WG espera solicitar que el Director avance este documento para la Recomendación Propuesta una vez que el Grupo de Trabajo tiene el apoyo demostrado por dos implementaciones interoperables. El Grupo de Trabajo, trabajando estrechamente con la comunidad de desarrolladores, espera mostrar estas implementaciones el 16 de febrero de 2007. El Grupo de Trabajo no planea solicitar avanzar a la Recomendación Propuesta antes del 16 de enero de 2007.

La segunda Última Llamada del Proyecto de Trabajo de esta especificación resultó una serie de comentarios de Última Llamada que han sido dirigidos por el grupo de trabajo.

Durante el primer periodo de Última Llamada de esta especificación, hubo comentarios insatisfechos que el grupo de trabajo fue incapaz de resolver de una manera que satisfaga al autor del comentario. El estado de estos comentarios no ha cambiado durante el segundo periodo de Última Llamada, como el grupo de trabajo continúa en pie por sus respuestas originales.

3.2.2. PRESENTE

El TT AF – DFXP proporciona una representación estandarizada de un subconjunto concreto de información textual con la que las semánticas estilísticas, de diseño y de sincronización están asociadas por un autor o un sistema de edición con el propósito de intercambio y la presentación potencial.

DFXP está expresamente diseñado para satisfacer sólo un conjunto limitado de requisitos establecidos. En particular, sólo aquellos requisitos que mantengan la necesidad de realizar el intercambio con los existentes, los sistemas de distribución de patrimonio están satisfechos.

Además de ser utilizado para el intercambio entre los formatos contenidos en la distribución del patrimonio, el contenido DFXP puede ser utilizado directamente como un formato de distribución.

En algunos contextos de uso, puede ser conveniente emplear contenido animado para describir las representaciones en lenguaje de signos del mismo contenido expresado por una instancia de documento de Texto Sincronizado. Este caso de uso no está explícitamente dirigido por los mecanismos DFXP, pero puede ser dirigido por alguna tecnología de integración multimedia externa.

3.2.3. MODELO DEL SISTEMA

El uso de DFXP está destinado para funcionar en un contexto más amplio de mecanismos de Edición y Distribución de Texto Sincronizado que están basados sobre un modelo de sistema, representado en la Figura 1 – Modelo del Sistema, en donde el TT AF sirve como un formato de intercambio bidireccional entre una colección heterogénea de los sistemas de edición, y como un formato de intercambio unidireccional a una colección heterogénea de formatos de distribución después de someterse a la transcodificación o compilación de los formatos de distribución de objetivos según se requiere, y donde un formato de distribución particular es DFXP.

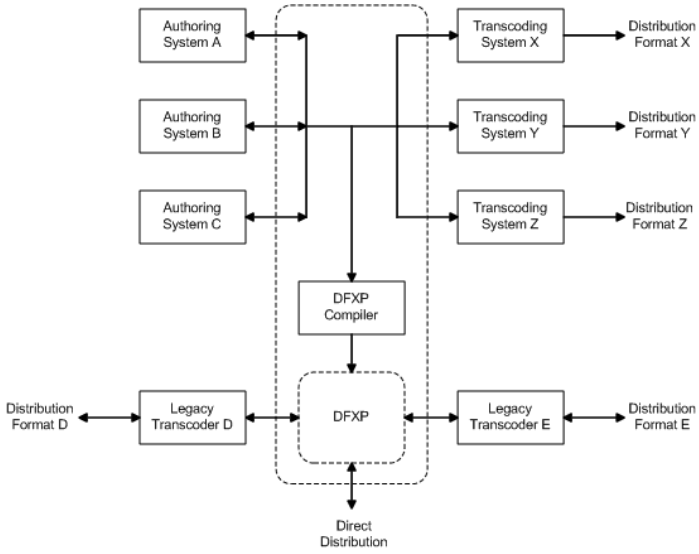

Figura 1 – Modelo del Sistema

3.2.4. DOCUMENTO DFXP

Una instancia de un documento DFXP consiste en un elemento *tt* del documento que contiene un encabezado y un cuerpo, donde la cabecera especifica el nivel de metadatos del documento, las definiciones de estilo y las definiciones de diseño, y el cuerpo especifica el contenido del texto entremezclado con referencias a la información de estilo y diseño e información de tiempo en línea.

El nivel de metadatos del documento puede especificar un título del documento, la descripción y la información del copyright. Además, los metadatos arbitrarios procedentes de otros espacios de nombres pueden estar especificados.

La información de estilo puede estar especificada en la forma de las definiciones de las especificaciones de estilo que están referenciadas por el diseño y la información de contenido.

La información de diseño define una o más regiones en las que el contenido está destinado a ser presentado. Una definición de región puede referenciar uno o más conjuntos de especificaciones de estilo para permitir que el contenido fluido en la región para heredar estos estilos.

El contenido de una instancia de un documento DFXP es expresado en su cuerpo, que es organizado en términos de bloque y elementos de texto en línea. La organización jerárquica de los elementos de contenido tiene un papel principal en la determinación de las relaciones espaciales y temporales.

3.2.5. CONFORMIDAD

Una instancia del documento TT AF debe cumplir los siguientes requisitos:

1. Cuando se transporte una instancia del documento en un contexto en el que un Tipo de los Medios de Comunicación MIME identifica el tipo de contenido de la instancia del documento intercambiado, entonces el tipo de medios de comunicación especificados es *application/ttaf+xml*, con el que un parámetro opcional *profile* puede aparecer.

2. La instancia del documento está o puede estar representada como un Conjunto de información Reducido XML.

3. El Conjunto de información Reducido XML que corresponde a la instancia del documento está o puede estar asociado con uno de los Tipos de Documento Resumen TT AF.

4. El Conjunto de información Reducido XML que corresponde a la instancia del documento es una instancia del Documento Resumen Válido de los Tipos de Documento Resumen asociados.

5. El Conjunto de información Reducido XML satisface todas las restricciones sintácticas y semánticas obligatorias y adicionales definidas. Además, este Conjunto de información debe satisfacer las directrices de accesibilidad para el contenido web.

Un procesador TT AF debe cumplir los siguientes requisitos:

1. El procesador proporciona al menos un mecanismo para la noción de instancias de una representación de un Conjunto de información Reducido XML de una instancia de documento conforme con TT AF.

2. Si un procesador hace o puede realizar la validación de una instancia de documento candidato TT AF, entonces proporciona al menos un mecanismo que asocia implícita o explícitamente la representación de un Conjunto de información Reducido XML de una instancia de documento conforme con TT AF con uno de los Tipos de Documento Resumen TT AF definidos.

3. El procesador, *a priori*, no rechaza o anula el procesamiento de una instancia de documento conforme con TT AF.

4. El procesador soporta todas las semánticas de procesamiento obligatorias definidas.

5. Si el procesador demanda soportar el procesamiento de presentación con el fin de producir una interpretación de un contenido TT AF en un medio visual, entonces deben estar implementadas la región y la semántica de la línea de diseño definidas. Además, el procesador debe satisfacer las pautas de accesibilidad agente de usuario especificadas.

6. Si el procesador soporta algunas semánticas de procesamiento opcionales definidas, entonces lo hace de una manera coherente con la semántica definida.

3.2.6. TIPOS DE DOCUMENTOS

Cada tipo de documento resumen se compone de las siguientes limitaciones:

- Una colección no vacía de tipos de elementos, donde cada tipo de elemento consta de un nombre, una colección (posiblemente vacía) de atributos, y una especificación de contenido.

- Una colección no vacía de tipos de elementos que pueden aparecer como el elemento del documento.

Una Instancia de Documento Resumen podrá ser evaluada en términos de validez, y es considerada para ser una Instancia de un Documento Resumen Válida si satisface la siguiente condición: si después

1. poda todos los temas de información de elemento cuyos nombres no son miembros de la colección de tipos de elementos definida por el tipo de documento resumen asociado, entonces

2. poda de la información de carácter del tema niños de cualquier elemento restante en caso de que todos los carácter niños del elemento denoten caracteres espacio en blanco XML y el tipo de elemento es definido como vacío en el Tipo de Documento Resumen asociado, y entonces

3. poda todos temas de información de atributo habiendo ampliado los nombres tales como el espacio de nombres URI de los nombres ampliados,

entonces el elemento de documento es uno de los tipos de elemento de documento permitidos por el tipo de documento resumen asociado, los descendientes del elemento de documento satisfacen sus tipos de elementos respectivos contenidos en las especificaciones, todos los atributos requeridos son presentados, y el valor declarado de cada atributo satisface el tipo declarado por el tipo de documento resumen asociado.

Mientras que un procesador conforme puede *a priori* no rechazar una instancia de documento conforme, una instancia de documento dada puede estar limitada por el autor o por la herramienta de edición a satisfacer una definición más restrictiva de validez.

El Contenido DFXP es un tipo de documento resumen del TT AF destinado para ser usado para el intercambio entre los sistemas de distribución.

El elemento (raíz) del documento de una instancia de documento DFXP debe ser un elemento *tt*.

3.2.7. ESPACIOS DE NOMBRES

El TT AF emplea una serie de espacios de nombres para elementos y ciertos atributos globales. La siguiente tabla especifica este conjunto de espacios de nombres e indica el prefijo por defecto usado dentro de esta especificación y la normativa URI que denota cada espacio de nombres.

En una instancia de documento específico, no es necesario que los prefijos por defecto mostrados abajo sean usados. Cualquier prefijo o nombre de espacios obligatorios que satisfacen las limitaciones de los espacios de nombres XML pueden ser usados que están asociados con los nombres de espacios URI especificados.

Nombre	Prefijo	Valor
TT	tt:	http://www.w3.org/2006/10/ttaf1
Parámetro TT	ttp:	http://www.w3.org/2006/10/ttaf1#parameter
Estilo TT	tts:	http://www.w3.org/2006/10/ttaf1#style
Extensiones de Estilo TT	ttsx:	http://www.w3.org/2006/10/ttaf1#style-extension
Metadatos TT	ttm:	http://www.w3.org/2006/10/ttaf1#metadata
Extensiones de Metadatos TT	ttmx:	http://www.w3.org/2006/10/ttaf1#metadata-extension

Tabla 1 – Espacios de Nombres

Si una referencia a un tipo de elemento es usada en esta especificación y el nombre del tipo de elemento no es espacio de nombres cualificado, entonces se aplica el Espacio de Nombres TT.

3.2.8. PERFILES

El TT AF emplea una serie de perfiles de su vocabulario y semánticas asociadas. La siguiente tabla especifica este conjunto de perfiles e indica un nombre normativo para cada perfil.

Nombre	Valor
DFXP	http://www.w3.org/2006/10/ttaf1#profile-dfxp

Tabla 2 – Perfiles

Una convención es definida para el uso por el autor del contenido para indicar el uso de un delta sustractivo o aditivo a un perfil predefinido para usar un sufijo *sub-perfil* opcional de un perfil URI: si el perfil es substractivo (resultado en un subconjunto propio), entonces el *sub-perfil* es expresado como "-token"; si es aditivo (resultado en un superconjunto propio), entonces es representado como "+token", donde *token* expresa un determinado autor de sub-perfil que se adhiere al tipo de datos *xsd:NCName* definido.

Un sistema de autoría TT AF puede indicar el perfil de TT AF usado en una instancia de documento por la especificación de un atributo *ttp:profile*. Un procesador de contenido TT AF puede hacer uso del valor de este perfil para asociar una instancia de documento con un esquema o funciones de procesamiento.

3.2.9. CATÁLOGO CENTRAL

El catálogo central define la línea de base, el vocabulario central del TT AF, y, en particular, el vocabulario de DFXP.

El catálogo de vocabulario central está destinado a satisfacer las necesidades de DFXP mientras proporciona un vocabulario de línea de base para futuros perfiles.

El vocabulario de elementos central especificado para usar con una instancia de documento TT AF es enumerado en la siguiente tabla:

Módulo	**Elementos**
Animación	set
Contenido	body, div, p, span, br
Documento	tt
Cabeza	head
Diseño	layout, region
Metadatos	Metadata
Temas de Metadatos	ttm:actor, ttm:agent, ttm:copyright, ttm:desc, ttm:name, ttm:title
Estilo	styling, style

Tabla 3 – Vocabulario de Elementos

Los grupos de vocabulario de elementos que son usados en los modelos de contenido definidos por los tipos de elementos TT AF son enumerados en la siguiente tabla:

Grupo	**Elementos**
Animation.class	set
Block.class	div | p
Inline.class	span | br | #PCDATA
Metadata.class	metada | ttm:agent | ttm:copyright | ttm:desc | ttm:title

Tabla 4 – Grupos de Vocabulario de Elementos

El vocabulario de atributos especificado para el uso con el catálogo de vocabulario central es enumerado en la siguiente tabla:

Módulo	**Atributos**
Atributos Centrales	xml:id, xml:lang, xml:space
Diseño	region
Atributos de Metadatos	ttm:agent, ttm:role
Atributos de Parámetros	ttp:cellResolution, ttp:clockMode, ttp:frameRate, ttp:frameRateMultipler, ttp:markerMode, ttp:pixelAspectRatio, ttp:profile, ttp:smpteMode, ttp:subFrameRate, ttp:tickRate, ttp:timeBase
Estilo	style
Atributos de Estilo	tts:backgroundColor, tts:color, tts:direction, tts:display, tts:displayAlign, tts:dynamicFlow, tts:extent, tts:fontFamily, tts:fontSize, tts:fontStyle, tts:fontWeight, tts:lineHeight, tts:opacity, tss:origin, tts:overflow, tts:padding, tts:showBackground, tts:textAlign, tts:textDecoration, tts:textOutline, tts:unicodeBidi, tts:visibility, tts:wrapOption, tts:writingMode, tts:zIndex
Atributos de Tiempo	begin, dur, end, timeContainer

Tabla 5 –Vocabulario de Atributos

3.2.10. CATÁLOGO DE EXTENSIONES

El catálogo de extensiones sirve como un marcador de posición para las extensiones del vocabulario central definido por DFXP.

El catálogo de extensiones está destinado para usar por los futuros perfiles del TT AF.

Dos espacios de nombres están reservados específicamente por la Tabla 1 – Espacios de Nombres para los metadatos y vocabulario de extensión:

- Extensiones de Estilo TT.

- Extensiones de Metadatos TT.

Además del vocabulario de extensión estandarizado, una instancia de documento TT AF conformado puede contener elementos cualificados de espacios de nombres arbitrarios que residen en cualquier otro espacio de nombres que aquellos espacios de nombres definidos para el uso. Además, una instancia de documento TT AF conforme puede contener atributos cualificados de espacios de nombres arbitrarios del vocabulario definido donde tales atributos residen en cualquier otro espacio de nombres que aquellos espacios de nombres definidos para el uso.

4. ARQUITECTURA

En este capítulo se describen los pasos seguidos en la realización de la aplicación. El primer paso es el planteamiento del problema así como la búsqueda de soluciones al mismo. El siguiente paso es el análisis y el diseño del la aplicación. El último paso es la implementación de dicha aplicación así como las pruebas para verificar el correcto funcionamiento de la misma.

4.1. PLANTEAMIENTO DEL PROBLEMA Y SOLUCIONES

Este apartado se corresponde con el primer paso en la realización de la aplicación, consistente en el planteamiento del problema que se trata en el presente PFC, así como de las soluciones propuestas para resolver dicho problema.

4.1.1. PLANTEAMIENTO DEL PROBLEMA

Mediante la realización del estudio de la documentación, así como, mediante las entrevistas mantenidas con el tutor, se lleva a cabo la determinación del problema que se presenta con la realización de este proyecto. Dicha determinación requiere un estudio sobre el incremento del uso del lenguaje HTML5 en detrimento del lenguaje Flash. Posteriormente, se realiza un estudio acerca de la carencia del uso de subtítulos en formato TT AF – DFXP existente en los visores de vídeos en formato HTML5 y, por tanto, la necesidad de implementar los subtítulos en dicho formato para mejorar la accesibilidad de los visores de vídeos en HTML5.

Se ha de tener en cuenta que el usuario final del sistema, es una persona con discapacidad auditiva por lo que hay que tener especial cuidado en diseñar una interfaz sencilla en su utilización, ya que este sector de la población tiene dificultades a la hora de interactuar con este tipo de tecnologías. Por tanto, se debe desarrollar una aplicación en la que se puedan llevar a cabo todas las actividades que ofrece, con la mayor simplicidad, sencillez, fiabilidad y eficacia como sea posible.

La interfaz de usuario es la parte de la aplicación con la que el usuario interactúa. Por lo tanto, es necesario tener en cuenta las discapacidades de las personas y adaptar la aplicación a estas discapacidades.

4.1.2. SOLUCIONES

Mediante el análisis y establecimiento de los requisitos que debe cumplir el software de la aplicación, se comenzará a proponer soluciones que resuelvan el problema.

Se debe realizar una aplicación de uso sencillo y fácil manejo, ya que este proyecto va dirigido principalmente hacia personas con discapacidad auditiva. Por lo tanto, es necesario realizar una evaluación completa de la aplicación con el fin de determinar si cumple con todos los requisitos y requerimientos establecidos.

Se define un ciclo de vida del proyecto que indica el orden en el que se deben realizar las tareas que lo componen. Para ello, se define el modelo en cascada del ciclo de vida de tal forma que el inicio de cada etapa debe esperar a la finalización de la inmediatamente anterior.

El modelo en cascada admite la posibilidad de hacer iteraciones, es decir, si se tiene que volver a una de las etapas anteriores al mantenimiento, hay que recorrer de nuevo el resto de etapas.

Las ventajas del modelo en cascada son las siguientes:

- La planificación es sencilla.

- La calidad del producto resultante es alta.

- Permite trabajar con personal poco cualificado.

El modelo en cascada está reflejado en la siguiente figura:

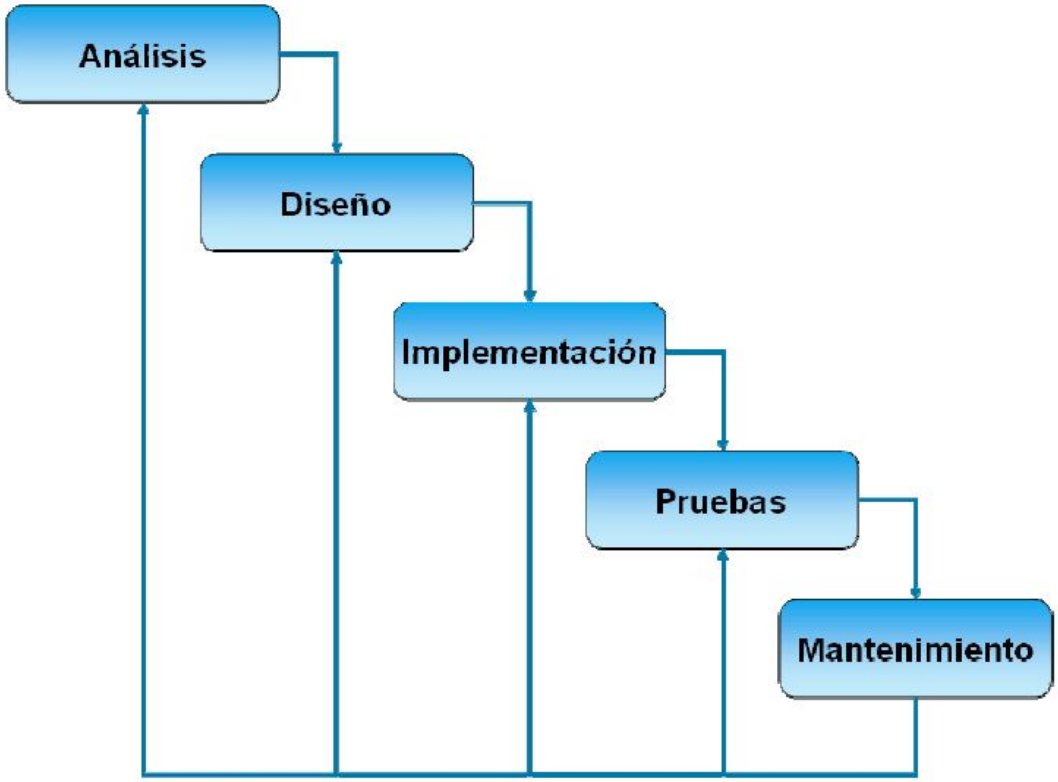

Figura 2 – Modelo en Cascada

El W3C en su *Iniciativa de Accesibilidad Web* proporciona un conjunto de pautas de accesibilidad para dar cabida a una variedad de discapacidades y deficiencias. Las siguientes pautas son referidas especialmente al grupo de las personas con discapacidad auditiva:

- Proporcionar alternativas equivalentes para el contenido visual y auditivo.

- Asegurar la accesibilidad directa de las interfaces de usuario incrustadas.

- Utilizar las tecnologías y pautas del W3C.

- Proporcionar información de contexto y orientación.

4.2. ANÁLISIS Y DISEÑO

Este apartado se corresponde con el segundo paso en la realización de la aplicación, consistente en el análisis de la aplicación, así como de su diseño.

El análisis establece con precisión las necesidades y condiciones de la aplicación a desarrollar. Esta fase es tomada como punto de partida para la realización del diseño de la aplicación.

Durante el diseño, se traducen los requisitos de la aplicación en una representación del software. El diseño debe implementar todos los requisitos especificados.

4.2.1. ANÁLISIS

A continuación, se explican los requisitos que debe tener la aplicación. El objetivo del análisis de requisitos es analizar y documentar las necesidades funcionales que deberán ser soportadas por la aplicación a desarrollar y obtener un conjunto de requisitos que debe cumplir el software de la aplicación. Se clasifican en dos grupos:

- *Requisitos funcionales*: son los que definen el comportamiento interno del software.

- *Requisitos no funcionales*: son los que especifican criterios que pueden usarse para juzgar la operación de un sistema.

Id	1	**Tipo**	RF
Descripción			
La aplicación va dirigida a las personas con discapacidad auditiva, por tanto, la información debe ser adaptada para este sector de la población.			

Tabla 6 – Requisito Funcional 1

Id	2	**Tipo**	RF
Descripción			
La aplicación permitirá al usuario poder ajustar el volumen del vídeo.			

Tabla 7 – Requisito Funcional 2

Id	3	**Tipo**	RF
Descripción			
La aplicación permitirá al usuario poder apagar el volumen del vídeo.			

Tabla 8 – Requisito Funcional 3

Id	4	**Tipo**	RF
Descripción			
La aplicación permitirá al usuario poder mostrar los subtítulos.			

Tabla 9 – Requisito Funcional 4

Id	5	**Tipo**	RF
Descripción			
Los subtítulos pueden cambiar de formato, tamaño, color, tipo de letra, posición dentro de la pantalla, etc.			

Tabla 10 – Requisito Funcional 5

Id	6	**Tipo**	RF
Descripción			
Los botones del visor de vídeos tendrán un texto asociado que indique claramente su significado.			

Tabla 11 – Requisito Funcional 6

Id	7	**Tipo**	RF
Descripción			
El uso del visor de vídeos será fácil y sencillo de recordar por el usuario.			

Tabla 12 – Requisito Funcional 7

Id	8	**Tipo**	RF
Descripción			
No se requiere ninguna autenticación para entrar en la aplicación.			

Tabla 13 – Requisito Funcional 8

Id	1	**Tipo**	RNF
Descripción			
La aplicación deberá funcionar en cualquier PC.			

Tabla 14 – Requisito No Funcional 1

Id	2	**Tipo**	RNF
Descripción			
La aplicación se construirá mediante el lenguaje HTML5.			

Tabla 15 – Requisito No Funcional 2

Id	3	**Tipo**	RNF
Descripción			
Los subtítulos deben seguir el estándar TT AF – DFXP.			

Tabla 16 – Requisito No Funcional 3

Id	4	**Tipo**	RNF
Descripción			
La aplicación debe funcionar para cualquier vídeo en formato *.OGV*.			

Tabla 17 – Requisito No Funcional 4

Id	5	**Tipo**	RNF
Descripción			
El texto descriptivo que acompaña a cada botón estará escrito en lengua española.			

Tabla 18 – Requisito No Funcional 5

Id	6	**Tipo**	RNF
Descripción			
Por defecto, el tipo de letra a utilizar en los subtítulos será *Sans Serif*.			

Tabla 19 – Requisito No Funcional 6

Id	7	**Tipo**	RNF
Descripción			
Por defecto, el tamaño de letra a utilizar en los subtítulos será 16 px.			

Tabla 20 – Requisito No Funcional 7

Id	8	**Tipo**	RNF
Descripción			
Por defecto, el color de letra a utilizar en los subtítulos será blanco.			

Tabla 21 – Requisito No Funcional 8

Id	9	**Tipo**	RNF
Descripción			
Por defecto, la alineación a utilizar en los subtítulos será centrada.			

Tabla 22 – Requisito No Funcional 9

Id	10	**Tipo**	RNF
Descripción			
Por defecto, los subtítulos estarán en *negrita*.			

Tabla 23 – Requisito No Funcional 10

4.2.2. DISEÑO

La fase de diseño sirve al desarrollador como guía para realizar una completa implementación de la aplicación.

Los objetivos de diseño son conseguir una interfaz gráfica accesible para personas con discapacidad auditiva, además de ser agradable para el usuario.

El visor de vídeos está diseñado dividiendo su espacio en dos partes:

- La parte principal que es donde se visualiza la reproducción del vídeo.

- Debajo de la parte principal se encuentran los botones del visor necesarios para poder interactuar con el mismo.

Se utilizan hojas de estilo CSS que permiten variar de forma rápida y sencilla la apariencia de la interfaz gráfica de esta aplicación.

La aplicación será identificada mediante un nombre característico para ser reconocida de una manera rápida y sencilla. El nombre de la misma será *Visor*.

El usuario podrá, cuando lo desee, iniciar o detener la reproducción del vídeo, también podrá aumentar o disminuir el volumen del mismo a su agrado, así como apagar el volumen y, por último, podrá mostrar o dejar de mostrar los subtítulos del vídeo.

El usuario podrá realizar todo lo descrito anteriormente haciendo clic sobre el botón correspondiente a cada caso.

4.3. IMPLEMENTACIÓN Y PRUEBAS

Este apartado se corresponde con el último paso en la realización de la aplicación, consistente en la implementación de la aplicación, así como de realizar las pruebas necesarias para verificar el correcto funcionamiento de la misma.

4.3.1. IMPLEMENTACIÓN

En esta fase se realiza la implementación de la aplicación, tomando como referencia el diseño realizado con antelación.

Como se ha comentado a lo largo del presente documento, el lenguaje de programación utilizado para implementar la aplicación es HTML5 y los subtítulos del vídeo están implementados siguiendo el estándar TT AF – DFXP.

La metodología de trabajo que marca el desarrollo de la aplicación contiene los siguientes pasos:

1. Creación de un fichero con extensión *.html* en HTML5 que será el principal y que contendrá la ruta en la que se encuentra el vídeo así como las llamadas a los ficheros necesarios, es decir, al fichero que contiene las funciones y al fichero que contiene los estilos.

2. Creación de un fichero Javascript con extensión *.js* que contendrá todas las funciones necesarias para el correcto funcionamiento de la aplicación. Desde este fichero se accederá al fichero con extensión *.xml* en el que se encuentran los subtítulos.

3. Creación de un fichero con extensión *.css* que es donde se definirán los estilos del visor así como los estilos que por defecto muestran los subtítulos.

4. Ejecución de la aplicación en el navegador.

En la siguiente figura se muestra de forma visual lo explicado anteriormente:

Figura 3 – Relación entre Ficheros

donde:

⟶ Indica que el fichero desde el que sale la flecha utiliza el fichero hacia el que llega la misma.

La aplicación dispone de una interfaz gráfica de usuario que permite a los clientes hacer clic en distintos botones del reproductor a la hora de reproducir el vídeo.

Figura 4 – Interfaz Gráfica

Como se puede observar en la anterior figura, lo primero que nos encontramos es la parte donde se visualiza el vídeo que se está reproduciendo. Debajo, se encuentran tanto los botones del reproductor como otros apartados que proporcionan información sobre la reproducción del vídeo.

Lo primero que nos encontramos es el botón *play*  que inicia la reproducción del vídeo y que, cuando se hace clic sobre él, se cambia por el botón *pause*  que detiene la reproducción del vídeo y que, cuando se hace clic sobre él, se cambia por el botón *play*.

Detrás, se encuentra la *barra de progreso*  en la que se visualiza el progreso que lleva el vídeo respecto al total.

Posteriormente, se encuentra un apartado 

en el que se muestra el tiempo actual por el que se está reproduciendo el vídeo y el tiempo total que dura el vídeo.

A continuación, se encuentra el botón de *apagar el volumen*  que apaga el volumen del reproductor.

Al lado, se encuentra el botón del *volumen*  que permite modificar el volumen del vídeo haciendo clic en cada una de las barras.

Por último, se encuentra el botón *subtítulos*  que permite mostrar los subtítulos y, si ya se están mostrando, permite dejar de mostrarlos.

La parte en la que se visualiza el vídeo se encuentra definida en el fichero *Visor.html* en donde se define el tamaño (ancho y alto) que tendrá por defecto al abrir dicho archivo en el navegador, también se hace referencia a los estilos utilizados y también se escribe la ruta en la que se encuentra el vídeo así como el tipo de vídeo. Todo esto se encuentra definido en dicho fichero de la siguiente forma:

```
<div class="caja-visor">
  <video class="visor" width="640" height="264" controls preload>
    <source src="firma6_flash8_700kbps.theora.ogv" type='video/ogg;
codecs="theora, vorbis"'>
  </video>
</div>
```

El resto de apartados del visor se encuentran definidos en el fichero *JavaScript.js* y sus respectivos estilos que están definidos en el fichero *hoja-de-estilos.css*. Como se ha comentado anteriormente, la llamada a estos ficheros tiene lugar desde *Visor.html* y se hace de la siguiente forma:

```
<script src="JavaScript.js" type="text/javascript" charset="utf-8"></script>
<link rel="stylesheet" href="hoja-de-estilos.css" type="text/css"
media="screen" charset="utf-8">
```

Los botones *play*, *pause*, *apagar volumen*, *volumen* y *subtítulos* así como para la *barra de progreso* necesitan que se haga clic sobre los mismos para realizar su respectiva función. Para ello, se ha utilizado en cada uno el método de javascript *addEventListener*.

```
this.video.addEventListener("play", this.enPlay.context(this), false);
this.subtitulosControl.addEventListener("click",
this.enSubtitulosControlClick.context(this), false);
```

Para crear la apariencia de cada uno de los botones del reproductor se utiliza el atributo *className* para hacer referencia al correspondiente estilo definido en la hoja de estilos, el atributo *innerHTML* en caso de que sea necesario que muestre algo por defecto por pantalla y el atributo *title* para mostrar el texto descriptivo de dicho botón al poner el ratón encima del botón correspondiente. También se ha utilizado el método de javascript *appendChild* para insertar el elemento correspondiente.

```
this.subtitulosControl = VVideo.crearElemento("li", { className: "visor-
subtitulos-control", innerHTML: "<span><b>S</b></span>", title:
"Subtitulos" });
this.controls.appendChild(this.subtitulosControl);
```

A medida que se va reproduciendo el vídeo, se actualiza el progreso del mismo que se visualiza en la *barra de progreso*. Para ello, hay que tener en cuenta el tiempo actual de reproducción del vídeo mediante *currentTime* así como la duración total del vídeo. Una vez actualizado dicho progreso, hay que actualizar el tiempo actual del vídeo que se muestra al usuario.

```
actualizacionPlayProgreso: function(){
  if (this.controls.style.display == 'none') { return; }
  this.playProgreso.style.width = ((this.video.currentTime / this.video.duration) * (VVideo.obtenerValorComputadoEstilo(this.soporteProgreso, "width").replace("px", ""))) + "px";
  this.actualizacionTiempoDisplay();
},
actualizacionTiempoDisplay: function(){
  this.tiempoActualDisplay.innerHTML = VVideo.formatoTiempo(this.video.currentTime);
  if (this.video.duration) { this.duracionDisplay.innerHTML = VVideo.formatoTiempo(this.video.duration); }
},
```

Cuando se cambie el volumen del vídeo, hay que actualizar el botón *volumen*. También hay que actualizarlo en caso de que el usuario haga clic sobre el botón *apagar volumen*, aparte de poner el volumen a 0.

```
actualizacionVolumenDisplay: function(){
  var numeroVolumen = Math.ceil(this.video.volume * 6);
  for(var i=0; i<6; i++) {
    if (i < numeroVolumen) {
      VVideo.agregarClase(this.volumenDisplay.children[i], "visor-volumen-subir-nivel");
    } else {
      VVideo.quitarClase(this.volumenDisplay.children[i], "visor-volumen-subir-nivel");
    }
  }
},
```

El tiempo que se muestra en el apartado del tiempo actual y tiempo total del vídeo, se obtiene del vídeo, pero viene en segundos con decimales, para lo que es necesario redondear dicho tiempo a segundos y convertirlo a formato *h:m:s* (horas, minutos y segundos) de modo que al usuario le resulte fácil entender el tiempo que se muestra.

```
formatoTiempo: function(secs) {
  var segundos = Math.round(secs);
  var minutos = Math.floor(segundos / 60);
  var horas = Math.floor(minutos / 60);
  minutos = Math.floor(minutos % 60);
  minutos = (minutos >= 10) ? minutos : "0" + minutos;
  segundos = Math.floor(segundos % 60);
  segundos = (segundos >= 10) ? segundos : "0" + segundos;
  return horas + ":" + minutos + ":" + segundos;
},
```

Como se viene mencionando a lo largo de la presente documentación, los subtítulos están conforme al estándar TT AF – DFXP. Durante las reuniones mantenidas con el tutor se ha acordado implementar una parte de dicho estándar, no en su totalidad.

Así pues, el Vocabulario de Atributos que se define en el Espacio de Nombres del Parámetro TT que está compuesto por *ttp:cellResolution*, *ttp:clockMode*, *ttp:frameRate*, *ttp:frameRateMultiplier*, *ttp:markerMode*, *ttp:pixelAspectRatio*, *ttp:profile*, *ttp:smpteMode*, *ttp:subFrameRate*, *ttp:tickRate* y *ttp:timeBase* no se ha tenido en cuenta a la hora de implementar la aplicación.

En cambio, el Vocabulario de Elementos del catálogo de vocabulario central que está compuesto por *tt*, *head*, *body*, *div*, *p*, *span* y *br* sí se ha implementado.

También se ha implementado *xml:id* correspondiente al Vocabulario de Atributos del catálogo de vocabulario central, pero no se han implementado *xml:lang* y *xml:space*. En caso de aparecer éstos últimos, la aplicación sigue con su ejecución ignorándolos.

Por otra parte, el Vocabulario de Atributos que se define en el Espacio de Nombres Estilo TT que está compuesto por *tts:backgroundColor*, *tts:color*, *tts:direction*, *tts:display*, *tts:displayAlign*, *tts:dynamicFlow*, *tts:extent*, *tts:fontFamily*, *tts:fontSize*, *tts:fontStyle*, *tts:fontWeight*, *tts:lineHeight*, *tts:opacity*, *tts:origin*, *tts:overflow*, *tts:padding*, *tts:showBackground*, *tts:textAlign*, *tts:textDecoration*, *tts:textOutline*, *tts:unicodeBidi*, *tts:visibility*, *tts:wrapOption*, *tts:writingMode* y *tts:zIndex* ha sido implementado excepto *tts:display*, *tts:dynamicFlow*, *tts:padding*, *tts:showBackground*, *tts:wrapOption* y *tts:writingMode*. En caso de aparecer los no implementados, la aplicación sigue con su ejecución ignorándolos.

Las Expresiones de Valor de Estilo *<alpha>*, *<color>*, *<digit>*, *<duration>*, *<familyName>*, *<genericFamilyName>*, *<flowFunction>*, *<flowIntervalFunction>*, *<hexDigit>*, *<integer>*, *<length>*, *<namedColor>*, *<quotedString>* y *<string>* no han sido implementadas.

Además, ha sido implementado el Vocabulario de Atributos relacionado con el tiempo compuesto por *begin*, *dur* y *end*.

El Vocabulario de Elementos que se define en el Espacio de Nombres de Metadatos TT compuesto por *ttm:title*, *ttm:desc*, *ttm:copyright*, *ttm:agent*, *ttm:name* y *ttm:actor* no ha sido implementado.

Aparte, el Vocabulario de Elementos que se define en el Espacio de Nombres de Metadatos TT compuesto por *ttm:agent* y *ttm:role* tampoco ha sido implementado.

Cuando el usuario hace clic sobre el botón *subtitulos* lo primero que se comprueba es si los subtítulos se están mostrando ya (en cuyo caso se dejan de mostrar) o no (en cuyo caso se muestran), para lo que se utiliza el *display* de los subtítulos.

```
this.subtitulosDiv.style.display = (this.subtitulosDiv.style.display == "none") ? "block" : "none";
this.subtitulosDivDos.style.display = (this.subtitulosDivDos.style.display == "none") ? "block" : "none";
```

Para mostrar los subtítulos, lo primero que hay que hacer es cargar el fichero XML que los contiene. Para ello, se llama a la siguiente función:

```
cargarXMLDoc: function(archivoXML){
  var xmlDoc;
  if (window.XMLHttpRequest){
    xmlDoc = new window.XMLHttpRequest();
    xmlDoc.open("GET", archivoXML, false);
    xmlDoc.send("");
    return xmlDoc.responseXML;
  } else if (ActiveXObject("Microsoft.XMLDOM")){
     xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
     xmlDoc.async = false;
     xmlDoc.load(archivoXML);
     return xmlDoc;
  }
  alert("Error cargando el documento.");
  return null;
},
```

Posteriormente, se obtienen los estilos y/o las regiones que estén definidas dentro de la etiqueta <*head*> de los subtítulos, para lo que se utiliza el método de javascript *getElementsByTagName*.

```
var regiones = xmlDoc.getElementsByTagName("tt")[0].getElementsByTagName("head")[0].getElementsByTagName("layout")[0].getElementsByTagName("region");
```

Una vez obtenidos los estilos y/o las regiones, se procede a analizar los párrafos (denominados con la etiqueta <*p*>) que contienen los subtítulos. Para ello, se llama a la función *construirSubtitulos* cada 200 milisegundos mediante *setInterval*.

```
setInterval(function(){ this.construirSubtitulos(parrafos, estilos, regiones);
}.context(this), 200);
```

En esta función, por cada párrafo, se obtiene el comienzo, así como la duración y/o el final del mismo, para lo que se utiliza el método de javascript *getAttribute*. Dependiendo del comienzo y final del párrafo, comparándolo con el tiempo actual del vídeo (mediante *currentTime*), se van mostrando unos subtítulos u otros.

```
var comienzo = parrafos[i].getAttribute("begin");
```

Dependiendo del número de líneas por párrafo que tengan los subtítulos y de si contienen o no la etiqueta <*span*>, se llama a la función correspondiente donde se analiza el párrafo y en la que se añade al *div* mediante innterHTML donde se muestran los subtítulos el correspondiente texto mediante *nodeValue*.

```
this.subtitulosDiv.innerHTML = par.firstChild.nodeValue;
```

Para añadir los estilos y/o regiones, se obtiene de cada párrafo el atributo correspondiente y se busca entre los estilos y/o regiones obtenidos anteriormente mediante el atributo *xml:id* y, posteriormente, se añaden a los subtítulos mediante la llamada a la función *anyadirEstilos*.

```
buscarEstilos: function(div, est, e){
  var j = 0;
  var encontrado = false;
  while (j < est.length && encontrado == false){
    var id = est[j].getAttribute("id");
    if (e == id){
      this.anyadirEstilos(div, est[j], est);
      encontrado = true;
    }
    j++;
  }
},
```

En la función *anyadirEstilos* se obtienen todos los estilos del Vocabulario de Atributos que se define en el Espacio de Nombres Estilo TT. Se comprueba los estilos que tiene cada párrafo y se añaden al mismo directamente, excepto los estilos *tts:extent* y *tts:origin* que tienen un tratamiento especial.

```
var colorFondo = est.getAttribute("tts:backgroundColor");
if (colorFondo != null){
  div.style.backgroundColor = colorFondo;
}
```

El estilo *tts:extent* está compuesto por dos números separados por un espacio en blanco, el primero de ellos se refiere al alto y el segundo al ancho.

```
var medida = est.getAttribute("tts:extent");
if (medida != null){
  var i = 0;
  var ancho = "";
  var alto = "";
  var encontrado = false;
  while (i < medida.length && encontrado == false){
    if (medida[i] == " "){
      encontrado = true;
    } else{
       ancho += medida[i];
    }
    i++;
  }
  while (i < medida.length){
    alto += medida[i];
    i++;
  }
  div.style.width = ancho;
  div.style.height = alto;
}
```

El estilo *tts:origin* también está compuesto por dos números separados por un espacio en blanco, el primero de ellos se refiere a arriba y el segundo a la izquierda.

A continuación, se muestra un ejemplo de fichero XML con subtítulos:

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<tt xml:lang="es" xmlns="http://www.w3.org/2006/04/ttaf1"
xmlns:tts="http://www.w3.org/2006/04/ttaf1#styling">
  <head>
    <layout>
      <region xml:id=" Default">
        <style tts:color="#FFFFFF" />
      </region>
      <region xml:id=" Julian">
        <style tts:color="#FFFF00" />
      </region>
    </layout>
  </head>
  <body>
    <div xml:lang="es">
      <p begin="0:00:35.06" end="0:01:05.00" region="Default">(La Donna
è mobile)</p>
      <p begin="0:01:05.80" end="0:01:08.17" region="Julian">(CANTA)
La donna è mobile...</p>
      <p begin="0:01:08.18" end="0:01:09.91" region="Julian">...qual
piuma al vento...</p>
      <p begin="0:01:09.96" end="0:01:12.02" region="Julian">...muta
d'accento...</p>
      <p begin="0:01:12.06" end="0:01:14.17" region="Julian">...e di
pensiero.</p>
    </div>
  </body>
</tt>
```

Ahora, se muestran algunos de los estilos utilizados para la confección tanto del visor de vídeos como de los subtítulos:

```
.caja-visor div.visor-subtitulos {
  position: absolute;
  width: 100%;
  bottom: 30px;
  text-align: center;
  font-family: sans-serif;
  font-weight: bold;
  font-size: 16px;
  color: #fff;
  padding-bottom: .5em;
}

.caja-visor ul.visor-controles {
  list-style: none;
  position: absolute;
  margin: 0;
  border: none;
  opacity: 0.85;
  color: #fff;
  display: none;
  left: 0;
  right: 0;
  height: 35px;
  padding-left: 35px;
  padding-right: 165px;
  padding-top: 0;
  padding-bottom: 0;
}

ul.visor-controles > li.visor-play-control {
  width: 25px;
  left: 0px;
}
```

4.3.2. PRUEBAS

En esta fase, se presentan las pruebas realizadas y se comprueba el correcto funcionamiento de la aplicación.

El objetivo que se pretende conseguir mediante la realización de estas pruebas es comprobar si los usuarios potenciales se sienten cómodos en el manejo de la aplicación. Por lo tanto, se ha de realizar una evaluación completa del sistema, ya que es fundamental que se verifique que la aplicación funciona correctamente y cumple todas las exigencias de los usuarios.

Llegados a este punto, nos encontramos en condiciones de elaborar una batería de pruebas, a través de las cuales se podrá comprobar el funcionamiento correcto de la aplicación:

- *Prueba 1*: haciendo clic en el botón *play* se inicia la reproducción del vídeo y dicho botón se cambia por el de *pause*. También se va actualizando el tiempo actual de reproducción del vídeo a medida que avanza el mismo. Así mismo, si se pone el ratón encima del botón, se muestra el nombre del mismo.

Figura 5 – Prueba 1

- *Prueba 2*: haciendo clic en el botón *pause* se detiene la reproducción del vídeo y dicho botón se cambia por el de *play*. El tiempo actual del vídeo también se detiene. Así mismo, si se pone el ratón encima del botón, se muestra el nombre del mismo.

Figura 6 – Prueba 2

- *Prueba 3*: haciendo clic en la *barra de progreso* se cambia el instante de reproducción del vídeo. Además, si se pone el ratón encima de la barra, se muestra el nombre de la misma.

Figura 7 – Prueba 3

- *Prueba 4*: haciendo clic sobre el botón *apagar volumen* se apaga el volumen del vídeo. Se puede observar que se ha apagado el volumen mirando en el botón del volumen. Además, si se pone el ratón encima del botón, se muestra el nombre del mismo.

Figura 8 – Prueba 4

- *Prueba 5*: haciendo clic en cada una de las barras del botón *volumen* se puede ajustar el volumen del vídeo. Además, si se pone el ratón encima del botón, se muestra el nombre del mismo.

Figura 9 – Prueba 5

- *Prueba 6*: haciendo clic en el botón *subtítulos* se muestran los subtítulos del vídeo. Si se hace clic de nuevo sobre dicho botón, los subtítulos se dejan de mostrar. Además, si se pone el ratón encima del botón, se muestra el nombre del mismo.

Figura 10 – Prueba 6

Las siguientes pruebas son cambios en los subtítulos del vídeo. Para hacer los cambios, hay que abrir el fichero XML en el que se encuentren los subtítulos y añadir donde se desee el atributo correspondiente.

- *Prueba 7*: se puede añadir color de fondo a los subtítulos, que por defecto no tiene. Para ello, hay que añadir *tts:backgroundColor*.

```
<p begin="0:00:35.06" end="0:01:05.80" tts:color="#FFFFFF"
tts:backgroundColor="black">(La Donna è mobile)</p>
```

Figura 11 – Prueba 7

- *Prueba 8*: se puede cambiar el color de letra de los subtítulos, que por defecto es blanco, por el color que se desee. Para ello, hay que añadir *tts:color*.

Hay muchas empresas <br/> que envían y reciben facturas...

Figura 12 – Prueba 8

- *Prueba 9*: se puede cambiar la direccionalidad de los subtítulos, es decir, poder escribirlos de derecha a izquierda ya que, por defecto, es de izquierda a derecha. Para ello, hay que añadir *tts:unicodeBidi* y *tts:direction*

¡Dios! ¡Qué porrazo me he 'pegao'!

Figura 13 – Prueba 9

- *Prueba 10*: se puede especificar la alineación de las zonas de bloqueo en la dirección de la progresión del bloqueo. Para ello, hay que añadir *tts:displayAlign*.

```
<p begin="0:11:34.67" end="0:11:36.80" tts:color="#00FFFF">
<span tts:color="#FFFFFF" tts:origin="200px 200px"
tts:extent="100px 20px" tts:displayAlign="before">Hola Pepa
</span><br/> <span tts:color="#00FFFF" tts:origin="225px
300px" tts:extent="100px 20px" tts:displayAlign="after"> Hola
Ana</span> </p>
```

Figura 14 – Prueba 10

- *Prueba 11*: se pueden cambiar las medidas (ancho y alto) de la región en la que se muestran los subtítulos. Para ello, hay que añadir *tts:extent.*

```
<p begin="0:11:47.64" end="0:11:49.64" tts:color="#FFFFFF"
tts:extent="330px 122px">(Risas)</p>
```

Figura 15 – Prueba 11

- *Prueba 12*: se puede cambiar el tipo de letra de los subtítulos, que por defecto es *Sans Serif*, por el tipo que se desee. Para ello, hay que añadir *tts:fontFamily*.

No pero es que...
  Que no.

Figura 16 – Prueba 12

- *Prueba 13*: se puede cambiar el tamaño de letra de los subtítulos, que por defecto es 16 px, por el tamaño que se desee. Para ello, hay que añadir *tts:fontSize*.

Que sea un cortadito
  Marchando un cortadito

Figura 17 – Prueba 13

- *Prueba 14*: se puede cambiar el estilo de los subtítulos por el que se desee. Para ello, hay que añadir *tts:fontStyle*.

```
<p begin="0:20:20.96" end="0:20:22.61" tts:color="#FFFFFF" tts:fontStyle="italic">Uyyy, ¡Pepa!</p>
```

Figura 18 – Prueba 14

- *Prueba 15*: se puede cambiar el peso de la fuente, que por defecto es negrita, por el peso que se desee. Para ello, hay que añadir *tts:fontWeight*.

```
<p begin="0:21:40.35" end="0:21:47.28" tts:color="#FFFFFF" tts:fontWeight="normal">(Música)</p>
```

Figura 19 – Prueba 15

- *Prueba 16*: se puede cambiar la distancia entre dos líneas de subtítulos. Para ello, hay que añadir *tts:lineHeight*.

¡Con lo fresquito <br/> que se tiene que estar allí!

Figura 20 – Prueba 16

- *Prueba 17*: se puede cambiar la opacidad de los subtítulos. Para ello, hay que añadir *tts:opacity*.

Pero, pero...
 ¡QUÉ MANAZAS ERES!

Figura 21 – Prueba 17

- *Prueba 18*: se puede cambiar la distancia entre la parte superior y la parte izquierda del visor y los subtítulos. Para ello, hay que añadir *tts:origin* con dos valores.

...a la delegación de Groenlandia...

Figura 22 – Prueba 18

- *Prueba 19*: se puede cambiar el desbordamiento de los subtítulos, es decir, si queremos que se muestre o no cuando ocupan más que el tamaño de la región. Para ello, hay que añadir *tts:overflow*.

Y encima se me va todo.

Figura 23 – Prueba 19

- *Prueba 20*: se puede cambiar la alineación de los subtítulos, que por defecto es centrada, por la alineación que se desee. Para ello, hay que añadir *tts:textAlign*.

```
<p begin="0:24:31.29" end="0:24:33.13" tts:color="#FFFFFF" tts:textAlign="end">Adiós a la confidencialidad.</p>
```

Figura 24 – Prueba 20

- *Prueba 21*: se puede cambiar la decoración de los subtítulos. Para ello, hay que añadir *tts:textDecoration*.

```
<p begin="0:24:12.34" end="0:24:14.34" tts:color="#FFFF00" tts:textDecoration="underline">...y yo puse 'Jota' de Josefa...</p>
```

Figura 25 – Prueba 21

- *Prueba 22*: se puede cambiar el esquema de los subtítulos. Para ello, hay que añadir *tts:textOutline*.

```
<p begin="0:05:39.37" end="0:05:40.54" tts:color="#FFFF00" tts:textOutline="red 2px 0px">¡Hombre, Adrián!</p>
```

Figura 26 – Prueba 22

- *Prueba 23*: se pueden ocultar los subtítulos sin necesidad de pulsar de nuevo el botón *subtitulos*. Para ello, hay que añadir *tts:visibility*.

```
<p begin="0:05:02.39" end="0:05:03.49" tts:color="#FFFF00" tts:visibility="hidden">¡¿A Groenlandia?!</p>
```

Figura 27 – Prueba 23

- *Prueba 24*: se pueden ordenar dos regiones diferentes de subtítulos en caso de que éstas se solapen. Para ello, hay que añadir *tts:zIndex*.

```
<p begin="0:06:14.23" end="0:06:16.01" tts:color="#FFFF00"
tts:origin="200px 200px" tts:extent="200px 25px"
tts:backgroundColor="black" tts:zIndex="0">¡Venga, hasta ahora!
<br/> <span tts:color="#FFFFFF" tts:origin="220px 260px"
tts:extent="200px 25px" tts:backgroundColor="blue" tts:zIndex="1">
¡Venga, hasta ahora!</span> </p>
```

Figura 28 – Prueba 24

Las siguientes pruebas tienen como finalidad comprobar que los estilos probados anteriormente se pueden utilizar juntos y no solamente de uno en uno. Se puede comprobar en la mayoría de pruebas anteriores puesto que para muchas ha sido necesario utilizar más de un estilo a la vez para que se viera claramente.

- *Prueba 25*: en este párrafo se cambian el color de fondo, el color, la dirección del texto y la alineación de los subtítulos. Para ello, se utilizan a la vez *tts:backgroundColor*, *tts:color*, *tts:direction*, *tts:unicodeBidi* y *tts:textAlign*.

...¡A EMPOLLARSE
 LA FIRMA DIGITAL!

Figura 29 – Prueba 25

- *Prueba 26*: en este párrafo se cambian el color, las medidas (ancho y alto) de la región que contiene los subtítulos y el tamaño de la fuente. Para ello, se utilizan a la vez *tts:color*, *tts:extent*, *tts:fontSize*.

¿¡PERO DÓNDE ESTABAS!?

Figura 30 – Prueba 26

- *Prueba 27*: en este párrafo se cambian el color, el estilo de la fuente y decoración del texto. Para ello, se utilizan a la vez *tts:color*, *tts:fontStyle* y *tts:textDecoration*.

```
<p begin="0:13:01.75" end="0:13:04.56" tts:color="#FF5A3F" tts:fontStyle="italic" tts:textDecoration="line-through">Segundooo, je, je, je.</p>
```

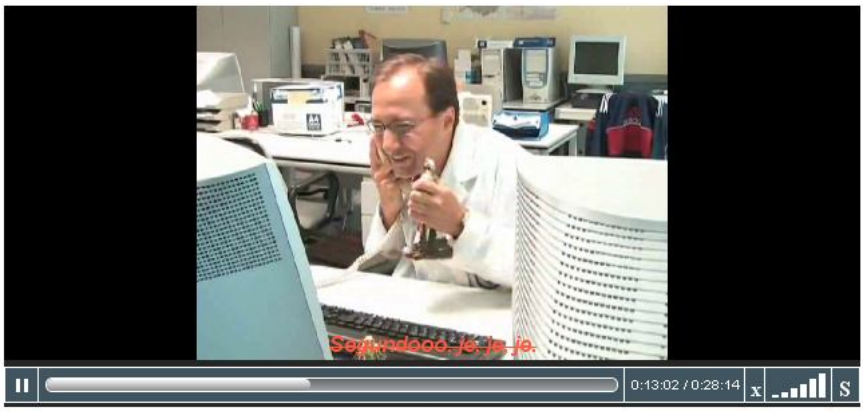

Figura 31 – Prueba 27

- *Prueba 28*: en este párrafo se cambian el color, el tipo de letra, la distancia entre dos líneas de subtítulos y opacidad. Para ello, se utilizan a la vez *tts:color*, *tts:fontFamily*, *tts:lineHeight* y *tts:opacity*. A diferencia de las anteriores pruebas, estos estilos se definen en la zona *region* definida dentro de la etiqueta <*head*>. En este caso, se ponen cada estilo dentro de una etiqueta <*style*> diferente.

```
<region xml:id=" Julian"> <style tts:color="#FFFF00" /> <style tts:fontFamily="times-new-roman" /> <style tts:lineHeight="32px" /> <style tts:opacity="0.75" /> </region>

<p begin="0:04:19.02" end="0:04:20.53" region="Julian">Y hay que asegurarse <br/> de que la recibe, ¿no?</p>
```

Figura 32 – Prueba 28

- *Prueba 29*: en este párrafo se cambian el color de fondo, las medidas (ancho y alto) de la región que contiene los subtítulos y el esquema de dichos subtítulos. Para ello, se utilizan a la vez *tts:backgroundColor*, *tts:extent* y *tts:textOutline*. A diferencia de la prueba anterior, se pueden poner todos los estilos dentro de la misma etiqueta <*style*>.

```
<region xml:id=" Pepa">  <style tts:backgroundColor="black" tts:extent="500px 70px" tts:textOutline="red 2px 0px"/>  </region>

<p begin="0:08:39.77" end="0:08:44.02" region="Pepa">¡¡JULIÁN!!, ¡¡JULIAN!! <br/> ¡¡CUÑADITO!!</p>
```

Figura 33 – Prueba 29

- *Prueba 30*: en este párrafo se cambian el tipo de letra, el tamaño, el estilo y la distancia entre la parte superior y la parte izquierda del visor y los subtítulos. Para ello, se utilizan a la vez *tts:fontFamily*, *tts:fontSize*, *tts:fontStyle* y *tts:origin*.

```
<region xml:id=" Vecina">  <style tts:fontFamily="times-new-roman" tts:fontSize="12px" tts:fontStyle="italic" tts:origin="50px 50px" />  </region>

<p begin="0:14:29.03" end="0:14:30.60" region="Vecina">La Fábrica Nacional <br/>  de Moneda y Timbre...</p>
```

Figura 34 – Prueba 30

- *Prueba 31*: en este párrafo se cambian el color, la alineación de las zonas de bloqueo en la dirección de la progresión del bloqueo, las medidas (ancho y alto) de la región que contiene los subtítulos, el peso de la fuente y la distancia entre la parte superior y la parte izquierda del visor y los subtítulos. Para ello, se utilizan a la vez *tts:color*, *tts:dispayAlign*, *tts:extent*, *tts:fontWeight* y *tts:origin*.

```
<p begin="0:05:41.71" end="0:05:42.94" tts:color="#FFFF00" tts:extent="150px 20px" tts:origin="200px 200px" tts:displayAlign="before" tts:fontWeight="normal">Qué bien me vienes. <br/> <span tts:color="#FFFFFF" tts:extent="150px 20px" tts:origin="225px 350px" tts:displayAlign="after" tts:fontWeight="normal"> ¿Cómo estamos?</span> </p>
```

Figura 35 – Prueba 31

- *Prueba 32*: en este párrafo se cambian el color de fondo, el color, las medidas (ancho y alto) de la región que contiene los subtítulos, la distancia entre la parte superior y la parte izquierda del visor y los subtítulos y el orden de dos regiones diferentes de subtítulos en caso de que éstas se solapen. Para ello, se utilizan a la vez *tts:backgroundColor*, *tts:color*, *tts:extent*, *tts:origin* y *tts:zIndex*.

```
<p begin="0:11:34.67" end="0:11:36.80" tts:color="#00FFFF">
<span tts:color="#FFFFFF" tts:origin="200px 200px" tts:extent="200px
25px" tts:backgroundColor="black" tts:zIndex="0">Hola Pepa
</span><br/> <span tts:color="#00FFFF" tts:origin="220px 260px"
tts:extent="200px 25px" tts:backgroundColor="blue" tts:zIndex="1">
Hola Ana</span> </p>
```

Figura 36 – Prueba 32

- *Prueba 33*: en este párrafo se cambian el color, la dirección del texto, las medidas (ancho y alto) de la región, el desbordamiento de los subtítulos, es decir, si queremos que se muestre o no cuando ocupan más que el tamaño de la región. Para ello, se utilizan a la vez *tts:color*, *tts:direction*, *tts:extent*, *tts:overflow*, tts:*unicodeBidi*.

```
<p begin="0:17:14.85" end="0:17:15.97" tts:color="#FFFFFF" tts:extent="100px 20px" tts:overflow="hidden" tts:unicodeBidi="bidi-override" tts:direction="rtl">Miro su certificado.</p>
```

Figura 37 – Prueba 33

- *Prueba 34*: en este párrafo se cambian el color, y la decoración del texto. Para ello, se utilizan a la vez *tts:color*, *tts:textDecoration*.

```
<p begin="0:24:50.83" end="0:24:51.72" tts:color="#FFFF00" tts:textDecoration="underline">Ya, ya.</p>
```

Figura 38 – Prueba 34

- *Prueba 35*: en este párrafo se cambian la distancia entre dos líneas de subtítulos, la alineación y el esquema de los subtítulos. Para ello, se utilizan a la vez *tts:lineHeight*, *tts:textAlign* y *tts:textOutline*.

```
<region xml:id=" Julian"> <style tts:lineHeight="26px" />
<style tts:backgroundColor="orange" /> <style tts:textAlign="start"
/> <style tts:textOutline="black 2px 0px" /> </region>

<p begin="0:02:08.51" end="0:02:11.53"
region="Julian">(ENFADADO) ¡Pepa! <br/> Otra vez te has 'olvidao'
las llaves.</p>
```

Figura 39 – Prueba 35

- *Prueba 36*: en este párrafo se cambian el color, el tipo de letra y el peso de la fuente. Para ello, se utilizan a la vez *tts:color*, tts*:fontFamily* y *tts:fontWeight*.

```
<region xml:id=" Informático"> <style tts:color="#FF5A3F"
tts:fontFamily="comic-sans" tts:fontWeight="normal" /> </region>

<p begin="0:12:44.93" end="0:12:45.96"
region="Informático">¿Buenas?</p>
```

Figura 40 – Prueba 36

- *Prueba 37*: en este párrafo se cambian las medidas (ancho y alto) de la región que contiene los subtítulos, el tamaño de la fuente y la opacidad. Para ello, se utilizan a la vez *tts:extent*, *tts:fontSize* y *tts:opacity*.

```
<region xml:id=" Cuñada">          <style tts:extent="550px 250px" />
<style tts:fontSize="18px" />          <style tts:opacity="0.75" />
</region>

<p begin="0:13:13.41" end="0:13:14.65" region="Cuñada">¡Ya va!</p>
```

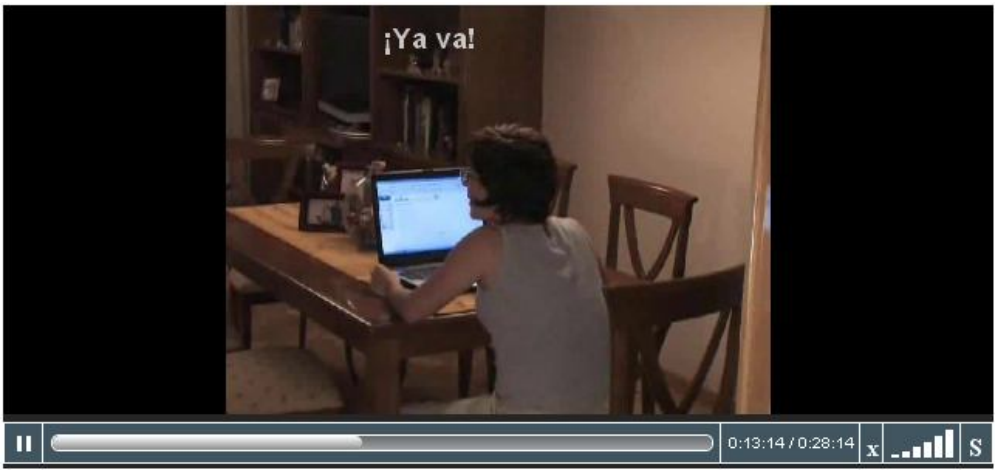

Figura 41 – Prueba 37

- *Prueba 38*: en este párrafo se cambian el color, el tamaño de la fuente y la opacidad. Para ello, se utilizan a la vez *tts:color*, *tts:fontStyle* y *tts:opacity*.

```
<region xml:id=" Cuñada">                    <style tts:color="#00FF00" />
<style tts:fontSize="20px" />                  <style tts:opacity="0.75" />
</region>

<p begin="0:21:01.58" end="0:21:03.85" region="Cuñada">¿Un ratón?</p>
```

Figura 42 – Prueba 38

- *Prueba 39*: con esta prueba se demuestra que los estilos que se definen en la etiqueta <*region*> se aplican a todos los párrafos que utilicen dicha región para añadir los estilos al subtítulo.

En este caso, se han añadido diferentes estilos a la región *Vecina* como son el color de fondo, el color, el estilo de la fuente y decoración del texto. Para ello, se utilizan a la vez *tts:backgroundColor*, *tts:color*, *tts:fontStyle* y *tts:textDecoration*.

```
<region xml:id=" Vecina">  <style tts:color="#FFFFFF" />
<style tts:backgroundColor="black" />  <style tts:fontStyle="italic"
/>  <style tts:textDecoration="underline" />  </region>
```

```
<p begin="0:13:22.67" end="0:13:25.14" region="Vecina">¡Uy hija!, te
has cortado el pelo  <br/> y además te has puesto gafas...</p>
```

Figura 43 – Prueba 39a

```
<p begin="0:13:25.26" end="0:13:26.52" region="Vecina">Oye, chica,
que te queda <br/>  mucho mejor...</p>
```

Figura 44 – Prueba 39b

- *Prueba 40*: con esta prueba se comprueba que se pasa de un párrafo al siguiente aunque el primero de ellos tenga el atributo *dur* (que corresponde con la duración del párrafo) y no el atributo *end* (que corresponde con el instante en el que termina el párrafo).

Pues nada, aquí, <br/> intentando desayunar.

Figura 45 – Prueba 40a

A ver... la diferencia entre el montado <br/> y la pulga, ¿es el tamaño?

Figura 46 – Prueba 40b

- *Prueba 41*: con esta prueba se comprueba que aparte de en las etiquetas <*p*> y en <*span*> (comprobado a lo largo de las anteriores pruebas), también se puede hacer referencia a los estilos o a las regiones en la etiqueta <*body*>.

```
<region xml:id=" Default">          <style tts:color="#FFFFFF" />
</region>

<body region="Default">
  <div xml:lang="es">
    <p begin="0:00:35.06" end="0:01:05.00">(La Donna è mobile)</p>
  </div>
</body>
```

Figura 47 – Prueba 41

- *Prueba 42*: con esta prueba se comprueba que, aparte de en las etiquetas <*p*>, <*span*> y <*body*>, se puede hacer referencia a los estilos o a las regiones desde la etiqueta <*div*>.

```
<region xml:id=" Jefe">                    <style tts:color="#FF0000" />
</region>

<div xml:lang="es" region="Jefe">
 <p begin="0:03:06.43" end="0:03:08.07" region="Jefe">¡Aysss!
Casi...</p>
</div>
```

Figura 48 – Prueba 42

- *Prueba 43*: con esta prueba se comprueba que en la etiqueta <*body*> también se pueden definir los estilos directamente en vez de hacer referencia a los estilos o a las regiones.

```
<body tts:color="#FFFF00">
  <div xml:lang="es">
    <p begin="0:01:05.80" end="0:01:08.17">(CANTA) La donna è mobile...</p>
  </div>
</body>
```

Figura 49 – Prueba 43

- *Prueba 44*: con esta prueba se comprueba que en la etiqueta <*div*> también se pueden definir los estilos directamente en vez de hacer referencia a los estilos o a las regiones.

```
<div xml:lang="es" tts:color="#FFFF00">
  <p begin="0:12:47.54" end="0:12:48.50">Eso dicen.</p>
</div>
```

Figura 50 – Prueba 44

- *Prueba 45*: con esta prueba se comprueba que aparte de en las etiquetas <*p*>, <*span*>, <*body*> y <*div*>, se puede hacer referencia a los estilos directamente desde la etiqueta <*br/*>.

```
<p begin="0:24:33.13" end="0:24:34.43" tts:color="#FFFF00">Y como se hizo un lío  <br tts:color="red" tts:backgroundColor="green" tts:fontStyle="italic"/> con las tarjetas...</p>
```

Figura 51 – Prueba 45

Las siguientes pruebas tienen como finalidad comprobar que los estilos se aplican correctamente a los subtítulos en caso de que en vez de la etiqueta <*region*> esté definida la etiqueta <*styling*>.

- *Prueba 46*: con esta prueba se comprueba que los estilos a los que se hacen referencia desde la etiqueta <*p*> se aplican correctamente.

```
<styling>
  <style id="Default" tts:color="#FFFFFF" tts:fontSize="20px"/>
</styling>

<p begin="0:05:55.77" end="0:05:58.79" style="Default">Por ejemplo, tenemos un sistema  <br/> fenomenal para las vacaciones.</p>
```

Figura 52 – Prueba 46

- *Prueba 47*: con esta prueba se comprueba que los estilos a los que se hacen referencia desde la etiqueta <*span*> se aplican correctamente.

```
<styling>
  <style id="Default" tts:color="#FFFFFF" tts:textAlign="end"/>
  <style id="Cuñada" tts:color="#00FF00" tts:fontFamily="verdana"/>
</styling>
```

```
<p begin="0:06:25.22" end="0:06:29.26" style="Cuñada">Ja ja ja ja
<br/> <span style="Default"> Ja ja ja ja</span> </p>
```

Figura 53 – Prueba 47

- *Prueba 48*: con esta prueba se comprueba que los estilos a los que se hacen referencia desde la etiqueta <*body*> se aplican correctamente.

```
<styling>
  <style id="Default" tts:color="#FFFFFF" tts:opacity="0.50"/>
<styling>

<body style="Default">
  <div xml:lang="es">
    <p begin="0:02:15.17" end="0:02:50.92">(Música)</p>
  </div>
</body>
```

Figura 54 – Prueba 48

- *Prueba 49*: con esta prueba se comprueba que los estilos a los que se hacen referencia desde la etiqueta <*div*> se aplican correctamente.

```
<styling>
 <style id="Julian" tts:color="#FFFF00" tts:backgroundColor="red"
tts:textOutline="black 2px 0px"/>
<styling>

<div xml:lang="es" style="Julian">
 <p begin="0:26:37.05" end="0:26:39.05">Es que son 20 horas de
vuelo.</p>
</div>
```

Figura 55 – Prueba 49

- *Prueba 50*: con esta prueba se comprueba que los estilos a los que se hacen referencia desde la etiqueta *
* se aplican correctamente.

```
<styling>
  <style id="Jefe" tts:color="#FF0000" tts:fontWeight="normal"/>
  <style id="Vecina" tts:color="#FFFFFF" tts:backgroundColor="black"
tts:textDecoration="line-through"/>
<styling>

<p begin="0:26:46.46" end="0:26:49.68" style="Jefe"
tts:color="#FF0000">Espera que te la presento, <br style="Vecina"/>
que está aquí mismo...pasa.</p>
```

Figura 56 – Prueba 50

5. PLANIFICACIÓN

En este capítulo se detallan las tareas que componen el proyecto, el tiempo empleado en la realización de cada una de ellas, así como el presupuesto del proyecto.

5.1. DESCOMPOSICIÓN EN TAREAS Y DURACIÓN

En este apartado se muestra la planificación de las tareas del proyecto, la duración de las mismas y la relación existente entre cada una de ellas a la hora de haber sido desarrolladas.

	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Inicio del proyecto	0 días	lun 04/10/10	lun 04/10/10	
2	**Desarrollo player HTML5 accesible**	**160 días**	**lun 04/10/10**	**vie 13/05/11**	
3	**Planteamiento del problema**	**20 días**	**lun 04/10/10**	**vie 29/10/10**	
4	Estudio del lenguaje HTML5	5 días	lun 04/10/10	vie 08/10/10	1
5	Estudio del estándar TT AF - DFXP	15 días	lun 11/10/10	vie 29/10/10	4
6	**Soluciones**	**5 días**	**lun 01/11/10**	**vie 05/11/10**	
7	Proponer diversas soluciones para resolver el problema	5 días	lun 01/11/10	vie 05/11/10	5
8	**Análisis**	**5 días**	**lun 08/11/10**	**vie 12/11/10**	
9	Análisis y especificación de requisitos	5 días	lun 08/11/10	vie 12/11/10	7
10	**Diseño**	**10 días**	**lun 15/11/10**	**vie 26/11/10**	
11	Diseño de la interfaz gráfica de usuario	10 días	lun 15/11/10	vie 26/11/10	9
12	**Implementación**	**75 días**	**lun 29/11/10**	**vie 11/03/11**	
13	Creación de la interfaz gráfica de usuario	15 días	lun 29/11/10	vie 17/12/10	11
14	Implementación del código fuente	60 días	lun 20/12/10	vie 11/03/11	13
15	**Pruebas**	**10 días**	**lun 14/03/11**	**vie 25/03/11**	
16	Realización de diversas pruebas	10 días	lun 14/03/11	vie 25/03/11	14
17	**Documentación del proyecto**	**35 días**	**lun 28/03/11**	**vie 13/05/11**	
18	Realización de la memoria	30 días	lun 28/03/11	vie 06/05/11	16
19	Realización de la presentación	5 días	lun 09/05/11	vie 13/05/11	18
20	Fin del proyecto	0 días	vie 13/05/11	vie 13/05/11	19

Figura 57 – Planificación de las Tareas del Proyecto

En la siguiente figura se ilustra de manera gráfica lo explicado anteriormente. Para ello, se ha realizado el *Diagrama de Gantt* correspondiente al proyecto.

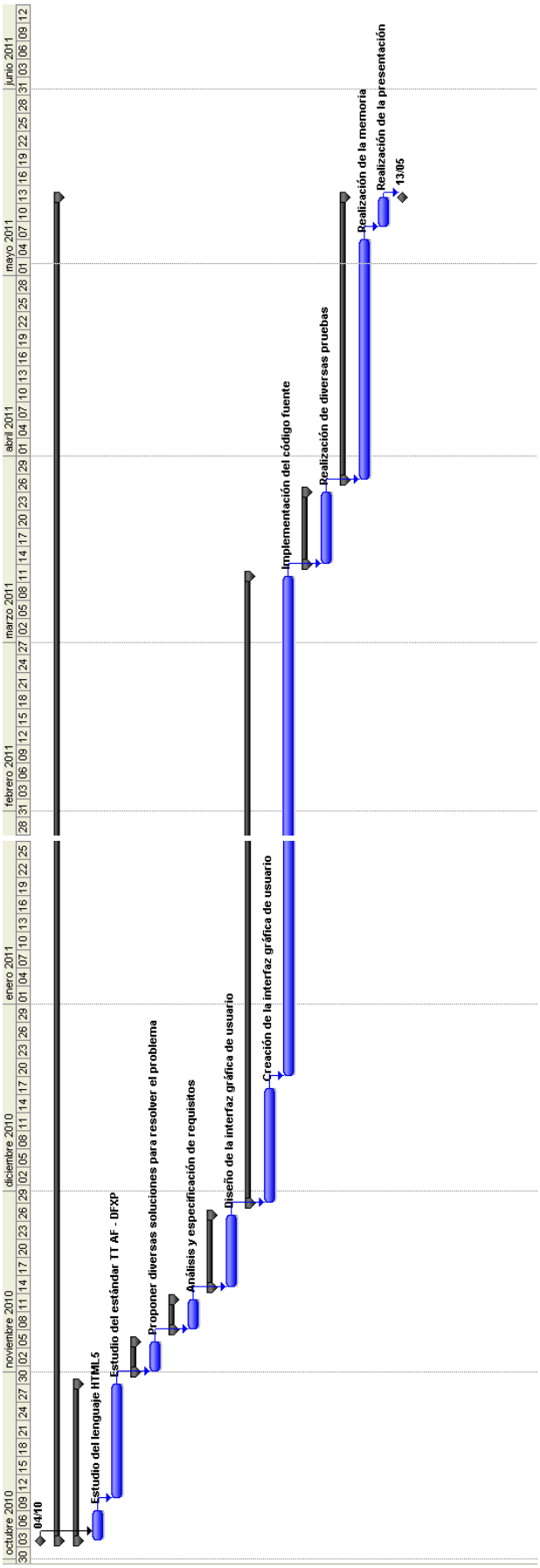

Figura 58 – Diagrama de Gantt del Proyecto

5.2. PRESUPUESTO

En este apartado se incluye el presupuesto total del proyecto dividido en recursos humanos y recursos materiales.

5.2.1. RECURSOS HUMANOS

Para el desarrollo de la aplicación ha sido necesario el uso de una persona, cuyo coste se desglosa según las actividades realizadas y previamente descritas y las horas dedicadas a cada una de las mismas. Se considera una retribución estimada de 40€/hora.

TAREA	COSTE/HORA (€)	HORAS	SUBTOTAL (€)
Estudio lenguaje HTML5	40	40	1.600
Estudio estándar TT AF – DFXP	40	120	4.800
Proponer diversas soluciones para resolver el problema	40	40	1.600
Análisis y especificación de requisitos	40	40	1.600
Diseño de la interfaz gráfica de usuario	40	80	3.200
Creación de la interfaz gráfica de usuario	40	120	4.800
Implementación del código fuente	40	480	19.200
Realización de diversas pruebas	40	80	3.200
Realización de la memoria	40	240	9.600
Realización de la presentación	40	40	1.600
TOTAL (€)			**51.200**

Tabla 24 – Coste Recursos Humanos

5.2.2. RECURSOS MATERIALES

Para el desarrollo de la aplicación también ha sido necesario el uso de material, cuyo coste se desglosa teniendo en cuenta el coste unitario de cada material y la cantidad que ha sido necesaria utilizar durante el desarrollo del proyecto.

CONCEPTO	COSTE/UD (€)	CANTIDAD	SUBTOTAL (€)
Ordenador sobremesa AMD Athlon 64 3.7GHz y 1GB RAM e impresora HP	1.200	1	1.200
Cartuchos impresora	30	2	60
Conexión a internet (coste mensual)	40	8	320
Macromedia Dreamweaver 8	300	1	300
Microsoft Office Professional 2007	300	1	300
Memoria USB 1GB	5	1	5
Papel A4 (500 hojas)	5	1	5
Encuadernación	5	3	15
Mozilla Firefox	0	1	0
TOTAL (€)			**2.205**

Tabla 25 – Coste Recursos Materiales

5.2.3. COSTE TOTAL

Por último, se calcula el coste total del proyecto, que se corresponde con la suma de los recursos humanos y los recursos materiales, a los que hay que añadir el 18% I.V.A.

CONCEPTO	**COSTE (€)**
Recursos humanos	51.200
Recursos materiales	2.205
Subtotal	53.405
I.V.A. (18%)	9.612,90
TOTAL (€)	**63.017,90**

Tabla 26 – Coste Total Proyecto

El presupuesto total de este proyecto asciende a la cantidad de SESENTA Y TRES MIL DIECISIETE EUROS CON 90 CÉNTIMOS DE EURO.

6. CONCLUSIONES

En este capítulo se recogen las principales conclusiones obtenidas tras la realización del presente proyecto.

Al comienzo del proyecto, el conocimiento que tenía sobre el tema de las diferentes discapacidades que tienen las personas era muy básico, así como el desarrollo de sistemas orientados a este grupo de la población. A medida que fue transcurriendo el proyecto, fui adquiriendo un gran conocimiento sobre este tema a través del estándar TT AF – DFXP, así como de otros documentos sobre la accesibilidad del W3C, que aumentaron mi interés a la hora de realizar este proyecto.

El motivo principal por el que quise realizar este proyecto fue el desarrollar un visor de vídeos en lenguaje HTML5 que de cada vez va a tener una mayor presencia en detrimento del lenguaje Flash y que, a su vez, fuese accesible a personas con discapacidad auditiva a través del estándar TT AF – DFXP algo que, cuando comencé el proyecto, no se había hecho.

Si comparamos los objetivos propuestos en el capítulo 2 de la presente memoria y lo que se ha logrado, podemos observar que se han cumplido todos los objetivos propuestos.

Recordemos que el objetivo principal del proyecto era el de realizar un visor de vídeos en lenguaje HTML5 accesible para personas con discapacidad auditiva.

Para la realización de este proyecto, un proceso clave ha sido el de la formación. Ha sido necesario obtener un profundo conocimiento para explotar al máximo las posibilidades tanto de los dispositivos como de las tecnologías empleadas.

Los subtítulos deben implementarse usando el estándar TT AF – DFXP que, al no estar familiarizado con él, buena parte del tiempo de desarrollo se ha dedicado a hacer un estudio sobre dicho estándar.

También ha sido necesario un estudio sobre el lenguaje HTML5, aunque para este estudio he necesitado menos tiempo que para el estudio del estándar TT AF – DFXP puesto que ya estaba familiarizado con el lenguaje HTML.

Por otra parte, todos y cada uno de los requisitos, tanto funcionales como no funcionales, definidos en la parte de análisis, se han cumplido.

Es importante llevar una buena organización en este tipo de proyectos, para lo cual se ha realizado la planificación inicial, porque, de lo contrario, la finalización del mismo se retrasaría bastante.

Con la finalización de este proyecto, se puede decir que no sólo nos encontramos satisfechos con el resultado obtenido, sino que también lo estamos con el trabajo realizado y el esfuerzo dedicado, siendo esto último quizás lo más importante para todos.

7. FUTURAS LÍNEAS/TRABAJOS

En este capítulo se comentan las posibles mejoras que se pueden realizar al proyecto.

Las futuras líneas/trabajos que se pueden realizar en este proyecto son las siguientes:

- Añadir un archivo de audio que permita hacer la aplicación accesible para personas con discapacidad visual. Para ello, es necesario añadir un botón al visor que permita al usuario activar esta opción cuando lo desee.

- Terminar de implementar todas las opciones recogidas en el estándar TT AF – DFXP y que no han sido implementadas en este proyecto tal y como se recoge en el apartado *4.3.1. Implementación* de la presente documentación.

- Añadir a los botones del visor, uno que permita al usuario poder visualizar el vídeo en formato pantalla completa.

- La aplicación funcione, aparte de en *Mozilla Firefox*, en el resto de navegadores, especialmente en los más utilizados que son *Internet Explorer* y *Google Chrome*, ya que, como se menciona en el apartado *Anexos* la aplicación no es compatible con el resto de navegadores.

ANEXOS

En este apartado se describen dos manuales para orientar tanto a los usuarios de la aplicación, para lo que se define el *Manual de usuario*, como para los posteriores desarrolladores que quieran ampliar el mismo, para lo que se define el *Manual de referencia*.

A. MANUAL DE USUARIO

En este manual, se procede a realizar una explicación de la utilización de la aplicación, que servirá al usuario como guía a la hora de interactuar con la aplicación.

A.1. COMPATIBILIDAD DE LA APLICACIÓN

Se ha comprobado la compatibilidad de la aplicación con los 3 navegadores más usados que son: *Internet Explorer*, *Mozilla Firefox* y *Google Chrome*.

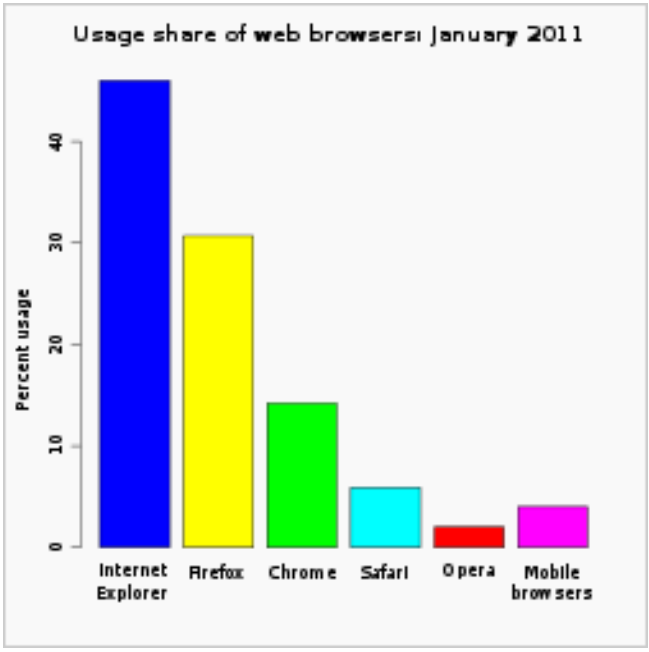

Figura 59 – Navegadores más utilizados en 2011

La aplicación no es compatible con los navegadores *Google Chrome* e *Internet Explorer*, pero sí que es compatible con el navegador *Mozilla Firefox*. Por tanto, el vídeo que se desee reproducir debe estar en formato *.OGV*.

Figura 60 – Icono Mozilla Firefox

Figura 61 – Icono Vídeo en formato .OGV

A.2. FUNCIONAMIENTO DE LA APLICACIÓN

Antes de empezar a usar la aplicación, hay que comprobar si vamos a reproducir el vídeo y los subtítulos que deseamos. Por lo tanto, es necesario abrir algunos ficheros, lo que se puede hacer simplemente con el *Bloc de notas*.

Primero, hay que abrir el fichero *Visor.html* y comprobar si la ruta escrita (se encuentra resaltado en la Figura 63 lo que hay que comprobar) corresponde con la ruta en la que se encuentra el vídeo. De no ser así, hay que actualizarla.

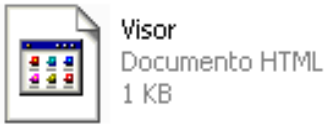

Figura 62 – Icono Visor.html

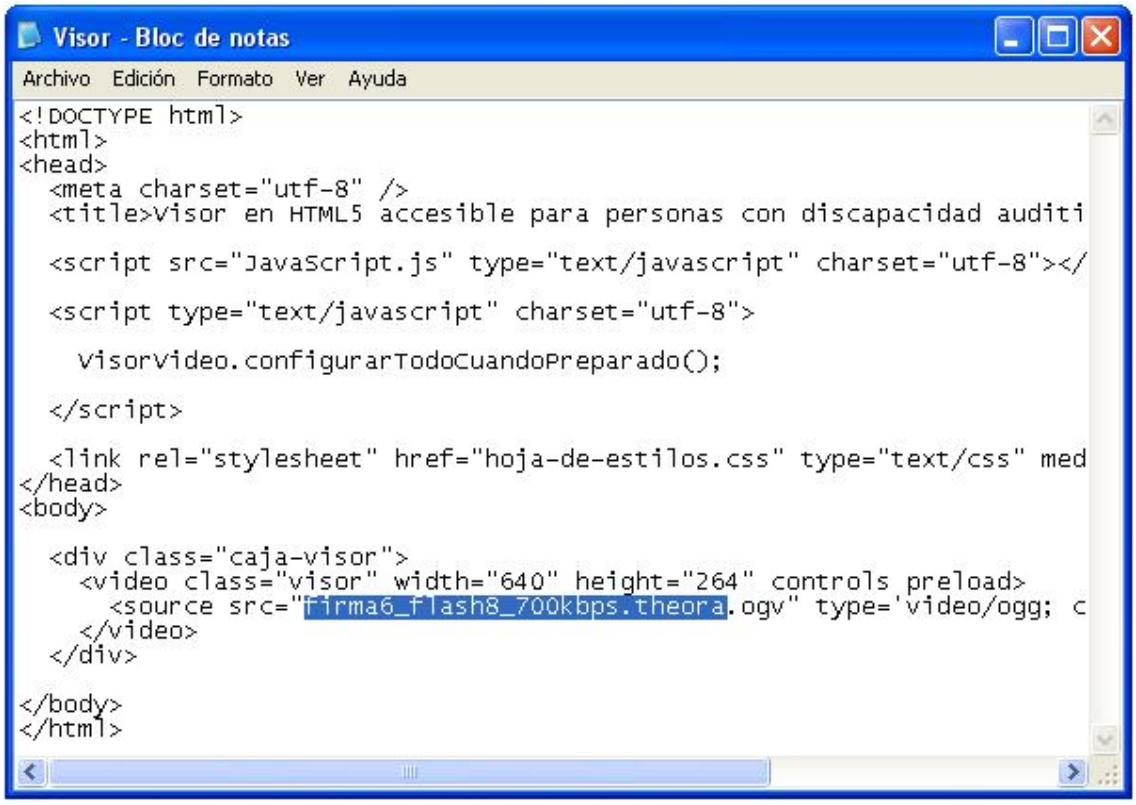

Figura 63 – Ruta vídeo

Después, hay que abrir el fichero *JavaScritp.js* y comprobar si la ruta escrita (se encuentra resaltado en la Figura 65 lo que hay que comprobar) corresponde con la ruta en la que se encuentra el archivo *XML* que contiene los subtítulos del vídeo. De no ser así, hay que actualizarla.

Figura 64 – Icono JavaScript.js

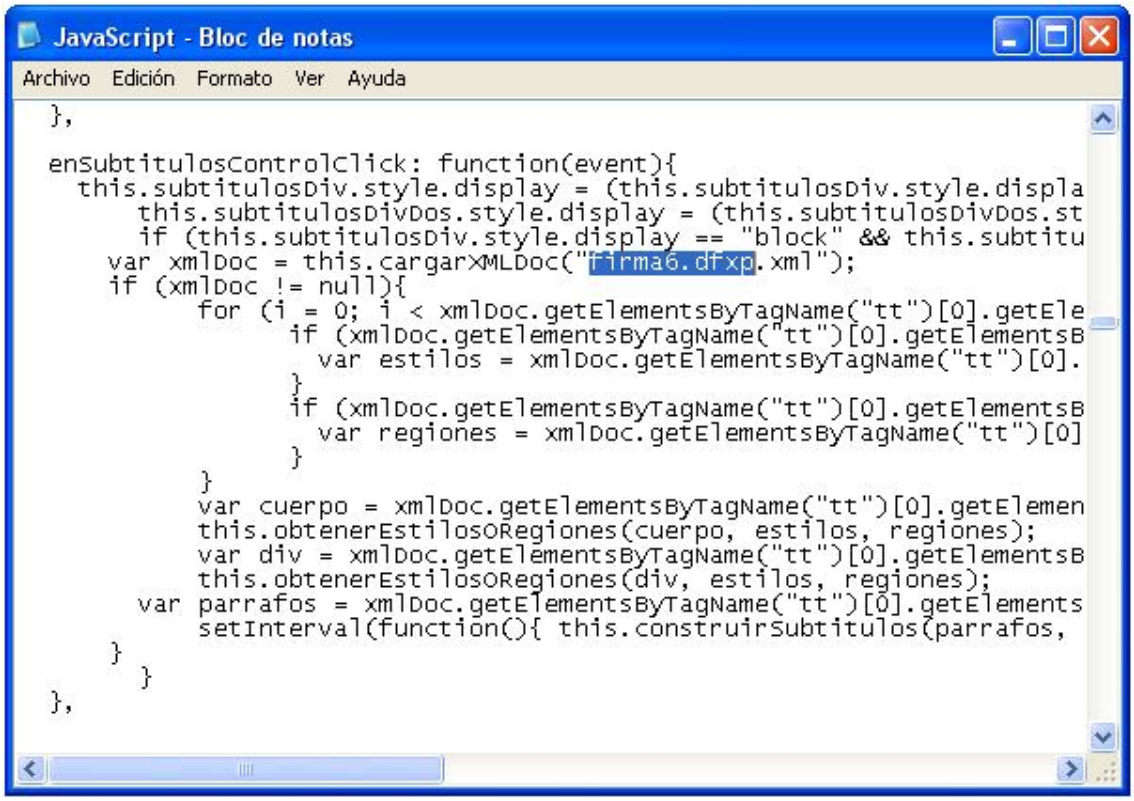

Figura 65 – Ruta subtítulos

Una vez comprobados tanto el vídeo como los subtítulos, hay que abrir el visor de vídeos. Para ello, como se ha mencionado anteriormente, se utiliza el navegador *Mozilla Firefox* y el archivo que hay que abrir con dicho navegador es *Visor.html*. Una vez abierto el visor, lo que se muestra por pantalla es lo siguiente:

Figura 66 – Visor de Vídeos

Componentes del reproductor:

- *Botón Play*: haciendo clic sobre este botón, se inicia la reproducción del vídeo y, en el reproductor, se cambia por el botón pause. Este botón se encuentra representado en la siguiente figura.

Figura 67 – Botón Play

- *Botón Pause*: haciendo clic sobre este botón, se detiene la reproducción del vídeo y, en el reproductor, se cambia por el botón play. Este botón se encuentra representado en la siguiente figura.

Figura 68 – Botón Pause

- *Barra de progreso*: en esta barra se visualiza el progreso que lleva el vídeo con respecto al total y haciendo clic sobre ella, se cambia el instante de reproducción del vídeo. Esta barra se encuentra representada en la siguiente figura.

Figura 69 – Barra de Progreso

- *Tiempo actual / tiempo total*: este apartado del reproductor nos muestra dos tiempos, el primero es el tiempo actual por el que se está reproduciendo el vídeo y el segundo es el tiempo total que dura el vídeo. El formato del tiempo mostrado es *h:m:s* (horas, minutos y segundos). Hacer clic en este apartado no produce ningún resultado. Este apartado se encuentra representado en la siguiente figura.

Figura 70 – Tiempo Actual / Tiempo Total

- *Botón apagar volumen*: haciendo clic sobre este botón, se apaga el volumen del vídeo. Este botón se encuentra representado en la siguiente figura.

Figura 71 – Botón Apagar Volumen

- *Botón volumen*: haciendo clic en cada una de las barras de este botón, se cambia el volumen del vídeo. Este botón se encuentra representado en la siguiente figura.

Figura 72 – Botón Volumen

- *Botón subtítulos*: haciendo clic sobre este botón, se muestran los subtítulos del vídeo. En caso de que se estén mostrando los subtítulos, al hacer de nuevo clic en este botón, se dejan de mostrar los mismos. Este botón se encuentra representado en la siguiente figura.

Figura 73 – Botón Subtítulos

- *Visualización del vídeo*: en este apartado, que se encuentra encima de todos los anteriores, es donde se visualiza el vídeo que se está reproduciendo. Hacer clic en este apartado no produce ningún resultado.

B. MANUAL DE REFERENCIA

En este manual, se procede a realizar una explicación técnica de la aplicación, que servirá para que cualquier desarrollador posterior pueda ampliar la aplicación.

B.1. COMPATIBILIDAD DE LA APLICACIÓN

Se ha comprobado la compatibilidad de la aplicación con los 3 navegadores más usados que, como se puede comprobar en la figura 59, son: *Internet Explorer*, *Mozilla Firefox* y *Google Chrome*.

La aplicación no es compatible con el navegador *Google Chrome* debido a que dicho navegador no es compatible con el formato XML, por lo que no es posible leer el fichero donde se encuentran los subtítulos que está en formato XML.

La aplicación tampoco es compatible con el navegador *Internet Explorer* debido a que dicho navegador no es compatible con el lenguaje HTML5, por lo que no es posible visualizar el reproductor de vídeos. La última versión de este navegador, *Internet Explorer 9*, sí que es compatible con HTML5, pero el inconveniente que presenta es que dicha versión solamente es compatible con el último sistema operativo lanzado por Microsoft, *Windows 7*.

La aplicación es totalmente compatible con el navegador *Mozilla Firefox* debido a que dicho navegador es compatible tanto con el lenguaje HTML5 como con el formato XML. Por tanto, el vídeo tiene que estar en formato *.OGV* para poder reproducirse correctamente.

B.2. FICHEROS UTILIZADOS

Los ficheros necesarios para el funcionamiento de la aplicación son los siguientes:

- *Visor.html*: en este fichero se escribe la ruta en la que se encuentra el vídeo que se desee reproducir así como las rutas de los ficheros javascript y CSS que son necesarios. Su icono está representado mediante la figura 62.

- *JavaScript.js*: en este fichero se definen todas las funciones necesarias para el correcto funcionamiento de la aplicación. Además, en la primera función de los subtítulos, se escribe la ruta en la que se encuentra el archivo *XML* que contiene los subtítulos del vídeo. Su icono está representado mediante la figura 64.

- *hoja-de-estilos.css*: en este fichero se definen todos los estilos necesarios. Su icono lo representa la siguiente figura:

Figura 74 – Icono hoja-de-estilos.css

- **.xml*: en este fichero se definen los subtítulos que tiene el vídeo en formato TT AF – DFXP. su icono lo representa la siguiente figura:

Figura 75 – Icono *.xml

- **.ogv*: en este fichero se encuentra el vídeo que se desea reproducir en la aplicación. Su icono está representado mediante la figura 61.

B.3. PROGRAMA UTILIZADO

La plataforma de desarrollo utilizada para implementar la aplicación ha sido *Macromedia Dreamweaver 8*.

En la siguiente figura, se muestra una vista de la ventana principal de Dreamweaver:

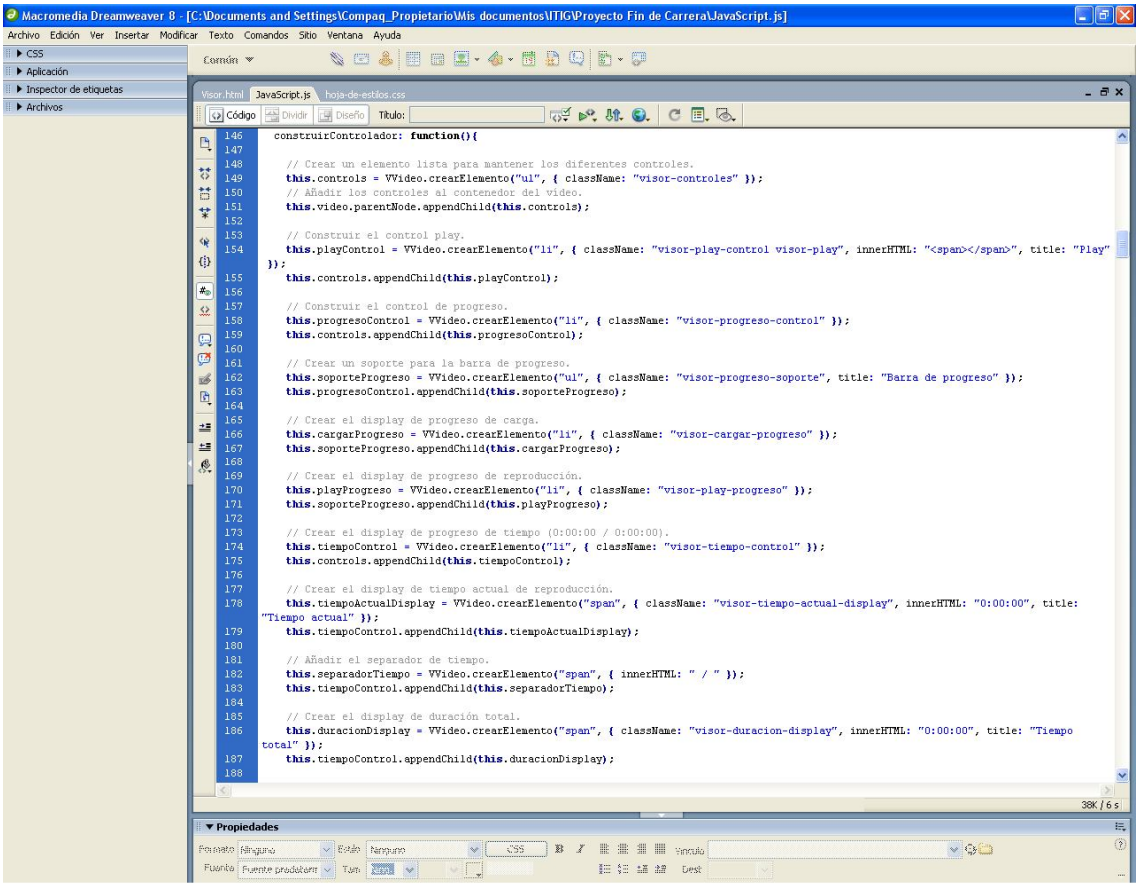

Figura 76 – Vista general Macromedia Dreamweaver 8

Se ha optado por utilizar Macromedia Dreamweaver 8 debido a que es la opción profesional para crear sitios Web y aplicaciones, dado que proporciona una potente combinación de herramientas visuales de diseño, funciones de desarrollo de aplicaciones y soporte para la edición del código, características todas ellas que permiten a los desarrolladores y diseñadores más expertos o menos expertos crear rápidamente sitios Web y aplicaciones basados en estándares.

Desde el avanzado soporte de diseño basado en CSS a las funciones de codificación manual, Dreamweaver proporciona las herramientas profesionales que requiere un entorno integrado y agilizado.

Los desarrolladores pueden utilizar Dreamweaver con su tecnología de servidor preferida para crear potentes aplicaciones en Internet destinadas a conectar a los usuarios a las bases de datos, las fuentes de datos dinámicos y los sistemas heredados.

GLOSARIO DE TÉRMINOS Y ACRÓNIMOS

API

Application Programming Interface

CERN

Conseil Européen pour la Recherche Nucléaire

CGI

Common Gateway Interface

DFXP

Distribution Format eXchange Profile

DOM

Document Object Model

DTD

Document Type Definition

HTML

HyperText Markup Language

IDL

Interface Definition Language

IETF

Internet Engineering Task Force

JSP

JavaServer Pages

RF

Requisito Funcional

RNF

Requisito No Funcional

RSS

Really Simple Syndication

SGML

Standard Generalized Markup Language

TT AF

Timed Text Authoring Format

TT WG

Timed Text Working Group

W3C

World Wide Web Consortium

WHATWG

Web Hypertext Application Technology Working Group

XHTML

eXtensible HyperText Markup Language

XML

eXtensible Markup Language

REFERENCIAS

En este apartado se definen las referencias utilizadas a la hora de la realización del proyecto así como de la documentación.

[1] Lenguaje HTML5
http://www.thinkepi.net/html5-nuevo-estandar-basico-del-web

[2] Antecedentes de HTML5
http://dev.w3.org/html5/spec/Overview.html

[3] Estándar TT AF – DFXP
http://www.w3.org/TR/2006/CR-ttaf1-dfxp-20061116/

[4] Ciclo de vida en cascada
http://www.ia.uned.es/ia/asignaturas/adms/GuiaDidADMS/node10.html

[5] Iniciativa de Accesibilidad Web del W3C
http://www.discapnet.es/web_accesible/wcag10/WAI-WEBCONTENT-19990505_es.html

[6] Definición de requisito funcional
http://es.wikipedia.org/wiki/Requisito_funcional

[7] Definición de requisito no funcional
http://es.wikipedia.org/wiki/Requisito_no_funcional

[8] Navegadores más utilizados en 2011
http://es.wikipedia.org/wiki/Guerra_de_navegadores

[9] Compatibilidad de Google Chrome con XML
http://techie-buzz.com/internet-tools/google-chrome-xml-support.html

[10] Internet Explorer 9 solamente compatible con Windows 7
http://windows.microsoft.com/es-ES/internet-explorer/products/ie/home

[11] Características Macromedia Dreamweaver 8
http://www.adobe.com/es/devnet/dreamweaver/articles/dw8_newfeatures.html